Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра систем искусственного интеллекта

УТВЕРЖДАЮ
Заведующий кафедрой
_____Г. М. Цибульский
«____ » _____ 2017 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

09.03.02.05 «Информационные системы и технологии в административном управлении»

Разработка автоматизированной информационной системы для отечественных легковых автомобилей «датчик дождя»

Руководитель _____ Д.А. Перфильев

Выпускник _____ С.Г. Кирсанов

Нормоконтролер _____ М.А. Аникьева

Красноярск 2017

Продолжение титульного листа бакалаврской работы по теме разработка автоматизированной информационной системы для отечественных легковых автомобилей «датчик дождя»

Нормоконтролер _____ М.А. Аникьева

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Информационные технологии и системы — класс дисциплин и отраслевых областей, относящихся к технологиям управления и обработки информации, в том числе, с применением вычислительных систем. Компонентами ИС являются данные, программное и аппаратное обеспечение.

Постоянное повышение уровня безопасности является первоочередной целью при разработке новых автомобилей. Важный вклад в обеспечение безопасности вносят новые вспомогательные системы для водителя, которые уже входят в комплектацию серийного автомобиля. По желанию автомобиль можно дооборудовать многими другими вспомогательными системами. Дождь и снегопад зачастую становятся огромной помехой безопасному управлению машиной, ведь водитель вынужден постоянно отвлекаться для того, чтобы включать-выключать щетки. Чтобы помочь водителю контролировать этот процесс, были придуманы датчики дождя. Они автоматически управляют стеклоочистителями. Одним из первых серийных автомобилей с датчиком дождя стал *Nissan Silvia 1994* года выпуска. В 2000-е годы датчики появились на автомобилях марок *Volkswagen, Cadillac, BMW* и других ведущих производителей. Если раньше этими устройствами оборудовались лишь дорогие машины, то сегодня, датчики могут устанавливаться в качестве опции на любой бюджетный автомобиль.

Цель бакалаврской работы: разработка автоматизированной информационной системы для отечественных легковых автомобилей «датчик дождя».

Для достижения цели необходимо решить следующие задачи:

1) Рассмотреть технологии информационных систем используемые в отечественных автомобилях;
2) Разработать автоматизированную информационную систему для отечественных легковых автомобилей «датчик дождя».

Глава 1. Информационные системы и их архитектуры, используемые в автомобилях

1.1 Анализ CAN-шины автомобиля

Еще в середине 80-х годов прошлого столетия компания *BOSCH* предложила новую концепцию сетевого интерфейса CAN (*Controller Area Network*). CAN-шина обеспечивает подключение любых устройств, которые могут одновременно принимать и передавать цифровую информацию (дуплексная система). Шины представляет собой витую пару. Данная реализация шин позволяет снизить влияние внешних электромагнитных полей, возникающих при работе двигателя и других систем автомобиля. По такой шине обеспечивается высокая скорость передачи данных. Компания *BOSCH* выполняет CAN шину по следующей схеме. На рисунке 1.1 приведен пример организации CAN шины фирмы *BOSCH*.

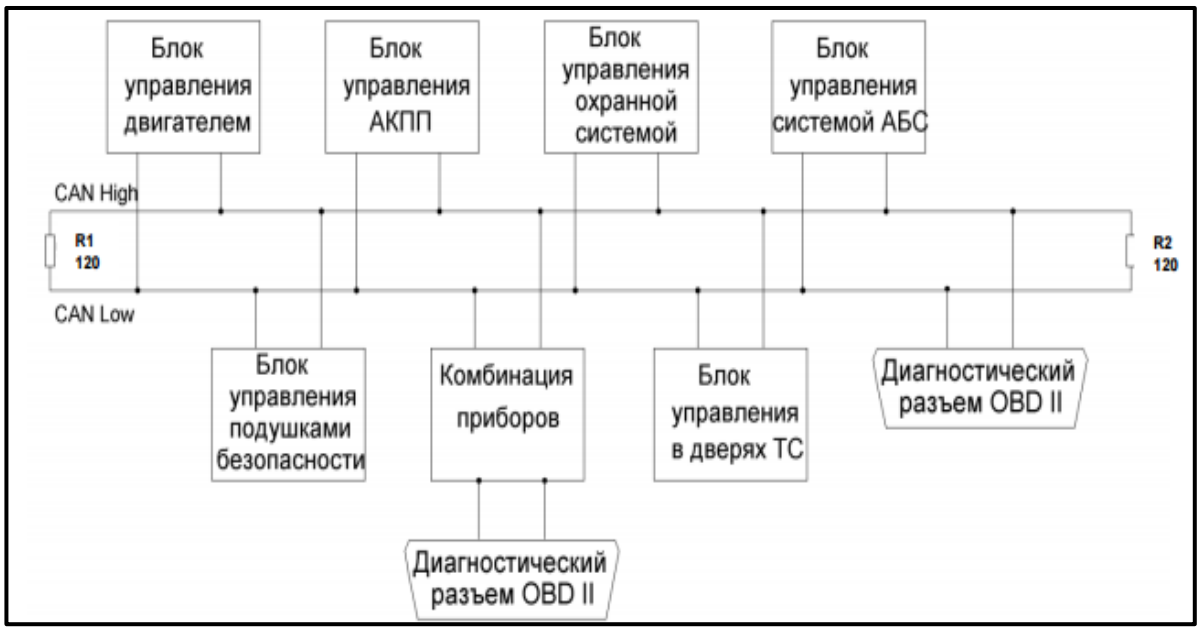


Рисунок 1.1 – Типовая схема шины CAN

Общая характеристика CAN-шины:

- интегрированная серийная коммуникационная шина для приложений работающих в режиме реального времени;

- сеть работоспособна при скорости обмена данными до 1 Мбит/с;

- обладает превосходными возможностями обнаружения и проверки ошибок и неисправностей;

- изначально CAN шина разработана для применения в автомобилях;

- используется в различных автоматических системах и системах управления.

Международный стандарт ISO 11898.

Топология CAN-шины:

- шина-CAN силового агрегата (быстрая шина), позволяющая передавать информацию со скоростью до 500 Кбит/с. Служит для связи между блоками управления на линии двигателя и трансмиссии, может находиться в доминантном состоянии при включенном зажигании;

- шина-CAN системы «Комфорт» (медленная шина), позволяющая передавать информацию со скоростью до 100 Кбит/с. Она служит для связи между блоками управления, входящими в систему «Комфорт» и прочих других, может находиться в доминантном состоянии при выключенном зажигании ТС;

- шина-CAN информационно-командной системы (медленная шина), позволяющая передавать информацию со скоростью до 100 Кбит/с. Служит для связи между различными обслуживающими системами. Может находиться в доминантном состоянии при выключенном зажигании ТС.

Системы и блоки управления автомобиля имеют не только различные нагрузочные сопротивления, но и скорости передачи данных, это может препятствовать обработке разнотипных сигналов. Для решения данной технической проблемы используется преобразователь для связи между шинами. Такой преобразователь называют межсетевым интерфейсом, это устройство в автомобиле чаще всего встроено в конструкцию блока управления,

комбинацию приборов, а также может быть выполнено в виде отдельного блока. На рисунке 1.2 представлена блок схема межсетевого интерфейса.

Рисунок 1.2 – Блок-схема межсетевого интерфейса

В CAN-шине архитектуры кольцо, соединяется последовательно несколько блоков управления, в том числе модуль управления двигателем (PCM), блок управления функциями салона (BCM) и блок управления топливным насосом. На выходе шины CAN предусмотрен «узел» для подключения блока управления полным приводом 4WD и оконечный резистор сопротивлением 120 Ом. В этой схеме все устройства равноправны, так как последовательно объединены в кольцо. Значит, передаваемые сигналы должны проходить по нескольким звеньям. Отсюда вытекают и недостатки схемы: потеря работоспособности при разрыве цепи или выходе из строя одного устройства; большая задержка и ее увеличение при добавлении нового звена.

На рисунке 1.3 представлена схема организации can шины по архитектуре «кольцо».

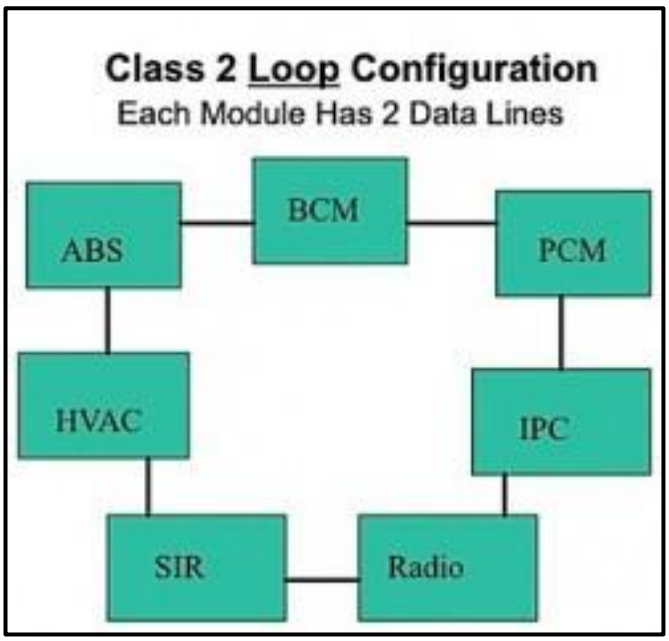


Рисунок 1.3 – Архитектура «Кольцо»

Следующий вид организации CAN-шины – это архитектура звезда. В такой архитектуре организации шины, все блоки подключены к одной точке. На рисунке 1.4 изображена схема организации can шины по архитектуре «звезда».

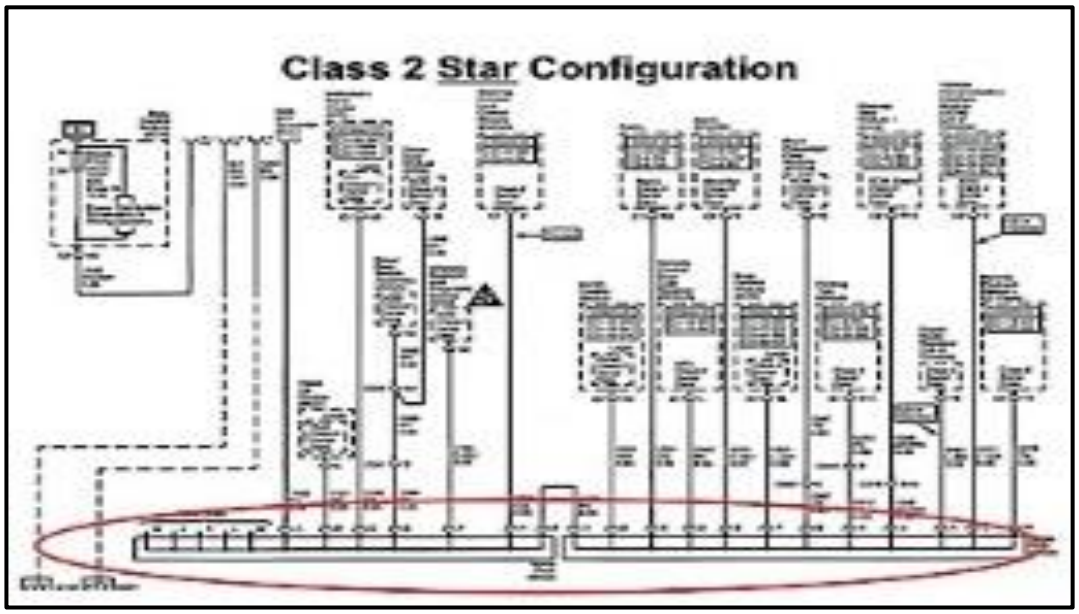


Рисунок 1.4 – Архитектура «звезда»

Также существует архитектура смешанного типа соединения, где комбинируются соединения типа «кольцо» и «звезда». На рисунке 1.5 представлена схема can шины смешанного типа соединения.

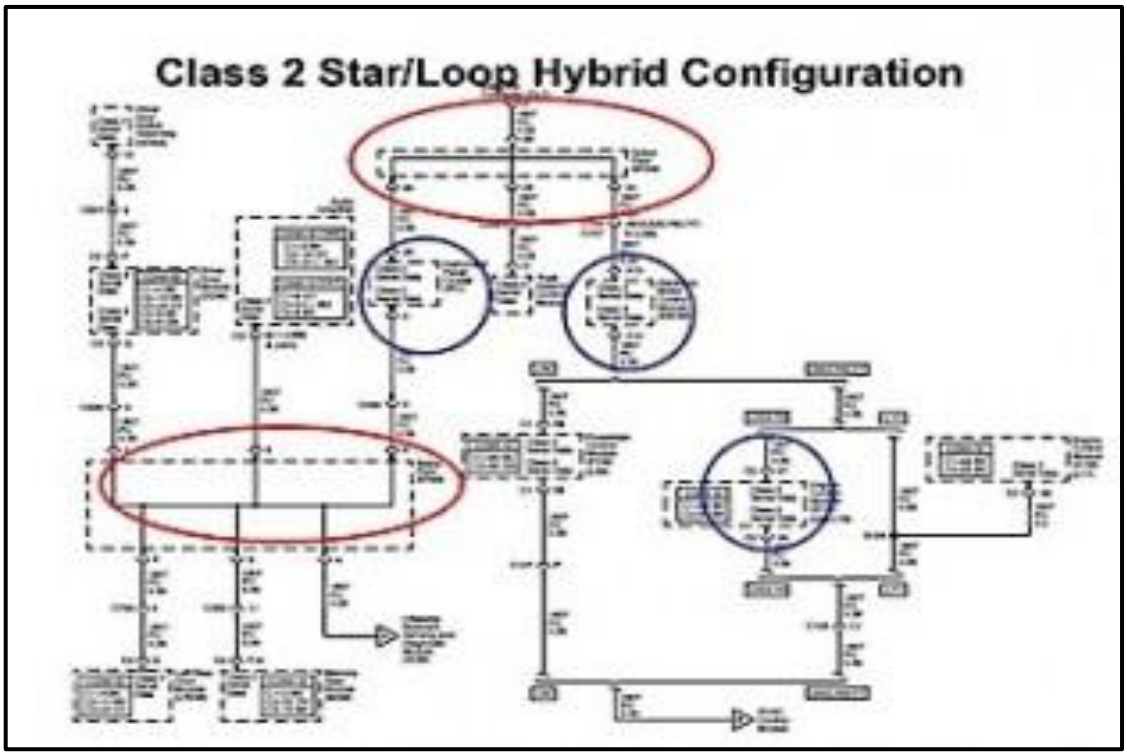


Рисунок 1.5 – Архитектура смешанного типа соединения

Окружностью красного цвета обведены точки сращивания или соединения звездой модулей ABS, противоугонной системы (*Theft Control*) и панели приборов. Кроме того, модуль ABS (обведен окружностью синего цвета) имеет двухпроводное подключение к шине и соединен с панелью приборов и диагностическим разъемом DLC по кольцевой схеме.

CAN стандарты:

- ISO 11898-1 - CAN протокол;

- ISO 11898-2 - CAN высокоскоростная физическая структура;

- ISO 11898-3 - CAN низкоскоростная физическая структура совместимая с ошибками;

- ISO 11898-4 - CAN запуск;

- ISO 11898-5 - Высокоскоростное низковольтное устройство;
- ISO 15765 - Диагностический протокол по CAN bus;
- ISO 11992 - определяет интерфейс тягачей и прицепов;
- J1939 Основной CAN протокол для грузовиков и автобусов.

1.2 Отечественные автоматизированные системы для автотранспорта

Волжский автомобильный завод перешел на использование бортовых компьютеров в 2002г. Первый автомобиль, который оснащался заводским бортовым компьютером АМК-211501, был ВАЗ-2114 в комплектации люкс. Амк-211501выпущен в 2002г. обрабатывает сигналы датчиков и индикации параметров движения автомобиля, расхода топлива, температуры окружающего воздуха, напряжения бортовой сети, временных параметров, диагностики электронных систем управления двигателем, а также для обнаружения препятствий и индикации расстояния до них при движении автомобиля задним ходом (с подключенным устройством парковки автомобиля). Компьютер устанавливается на автомобили ВАЗ 2108, ВАЗ 2109, ВАЗ-21099, ВАЗ-2115 с двигателем карбюраторного типа или оснащенных ЭСУД с электронным блоком управления. Компьютер имеет корпус, на передней части которого установлена панель с жидкокристаллическим индикатором и десятью клавишами для управления компьютером. На задней стенке корпуса расположен разъем для подключения жгута автомобиля. Семь клавиш служат для выбора нужной группы функций и выбора функций внутри группы и имеют следующее обозначение: « T », « КМ/Н », « КМ », « L », « L/100 », « ECU », « Н ». Клавиша « START » предназначена для определения начала поездки и сброса накапливаемых параметров, для фиксирования значения параметра в режиме коррекции и для установки или снятия режима контроля параметра. Клавиши « + » и « - » служат для увеличения или уменьшения значения параметра в

режиме коррекции параметров и для просмотра диагностической информации. На рисунке 1.6 показан внешний вид и кнопки управления Амк-211501.

Рисунок 1.6 – Внешний вид АМК211501

Исходя из отзывов пользователей *Lada-forum.ru* АМК211501 имеет ряд недостатков, а именно неточность показания вольтметра бортовой сети примерно на 0,2 В, также выдает не все коды ошибок неисправностей узлов автомобиля.

Бортовой компьютер *Lada Kalina* выпускается с 2006 г. выводит на экран 9 показаний - одно, в верхней строке, показывается постоянно, остальные, в нижней, можно выбирать. На рисунке 1.7 изображен БК *Lada Kalina*.

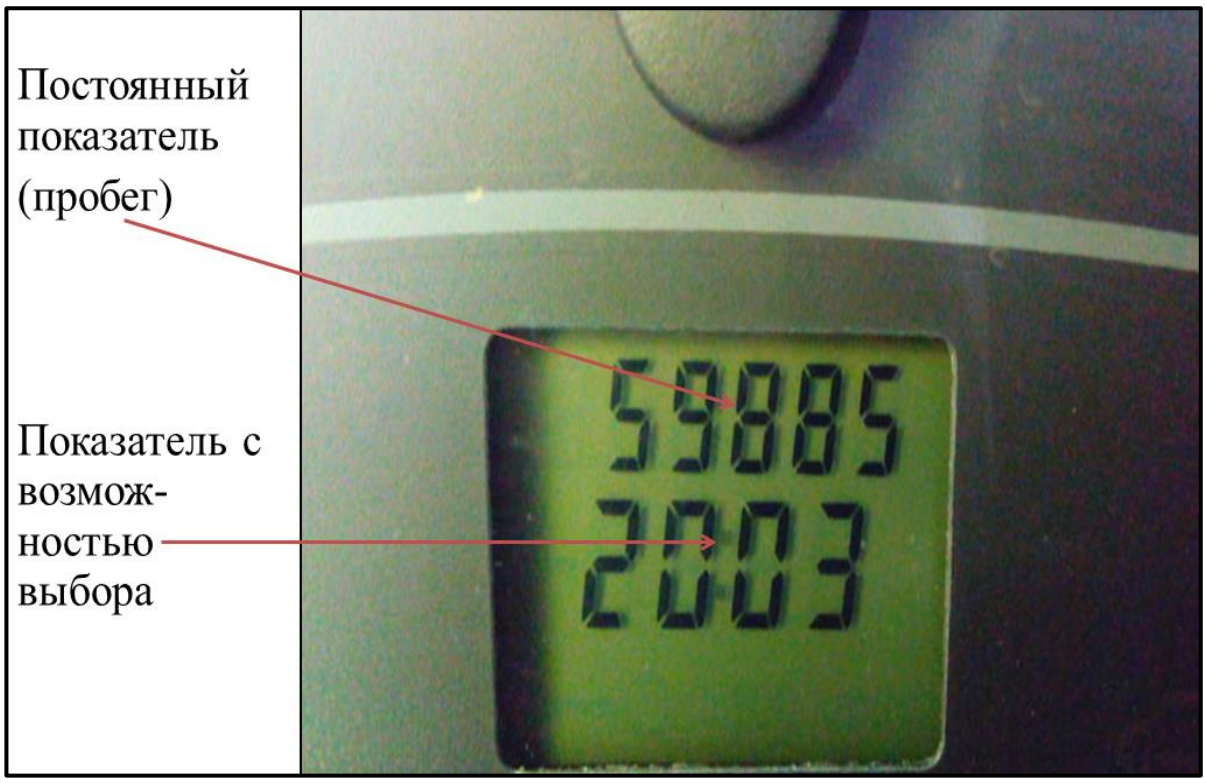


Рисунок 1.7 – Внешний вид бортового компьютера *Lada Kalina*

Функции бортового компьютера *Lada Kalina*:

- показывает текущее время в 24-часовом формате;
- израсходованное топливо во время последней поездки;
- средняя скорость движения во время последней поездки;
- запас хода;
- мгновенный расход топлива;
- средний расход топлива;
- температура воздуха за бортом.

Для переключения между показателями на правом подрулевом переключателе находится кнопка, которую можно нажимать как вверх, так и вниз, а на дисплее будут меняться режимы. Для сброса таких параметров, как средний расход топлива, время в пути, пробег с последнего запуска, а также настройки текущего времени — используется кнопка RESET, которая

находится на этом же переключателе снизу. На рисунке 1.8 представлен подрулевой переключатель управления БК *Lada Kalina*.

Рисунок 1.8 – Кнопка управления бортового компьютера *Lada Kalina*

Главный недостаток бортового компьютера по мнению пользователей *Drive2.ru* – это отсутствие показателя температуры двигателя автомобиля.

Бортовой компьютер *Lada Vesta* выпускается с 2016 г. и имеет ряд полезных функций, которых нет в бортовых компьютерах многих автомобилей, например, можно узнать напряжение бортовой сети или ограничитель скорости. На рисунке 1.9 представлен внешний вид БК *Lada Vesta*.

Рисунок 1.9 – Внешний вид бортового компьютера *Lada Vesta*

Функции бортового компьютера:

- счетчик общего пробега;
- счетчик пробега за поездку;
- время;
- время в пути;
- средняя скорость;
- подсказка переключения передач;
- напряжение бортовой сети;
- текущий расход топлива;
- средний расход топлива;
- расход топлива за поездку;
- запас хода;
- работа амт и номера включения передачи;

- температура окружающего воздуха;

- индикация ограничения скорости;

- индикация скорости круиз-контроля;

- включение/выключение звуковой подсказки переключения передач.

Выбор функций бортового компьютера осуществляется при помощи подрулевого переключателя аналогично бортовому компьютеру *Lada Kalina*.

Согласно мнению пользователей веб ресурса лада.онлайн.рф, бортовой компьютер *Lada Vesta* имеет больший функционал в сравнении с конкурентами, а именно встроенный вольтметр бортовой сети и ограничитель скорости. Главный недостаток отсутствие показателя температуры двигателя.

1.3 Микроконтроллер и его программирование

Семейство *ARM Cortex* - новое поколение процессоров, которые выполнены по стандартной архитектуре и отвечают различным технологическим требованиям. В отличие от других ЦПУ *ARM,* семейство *Cortex* является завершенным процессорным ядром, которое объединяет стандартное ЦПУ и системную архитектуру. Семейство *Cortex* доступно в трех основных профилях:

- профиль A для высокопроизводительных применений;

- профиль R для реально-временных применений;

- профиль M для чувствительных к стоимости и микроконтроллерных применений.

Микроконтроллеры *STM32* выполнены на основе профиля *Cortex-M3*, которое специально разработано для применений, где необходимы развитые системные ресурсы и малое энергоптребление. Они характеризуются настолько низкой стоимостью, что могут конкурировать с традиционными восьми и 16-битными микроконтроллерами. ЦПУ *ARM7* и *ARM9* были с успехом интегрированы в стандартные микроконтроллеры, в них все же прослеживается изначальная ориентированность на системы на кристалле (SoC). Это особенно

заметно по способам обработки исключительных ситуаций и прерываний, т.к. у разных производителей микроконтроллеров и способы обработки реализованы различным образом. *Cortex-M3* является стандартизованным микроконтроллерным ядром, которое помимо ЦПУ, содержит все остальные составляющие основу микроконтроллера элементы (система прерываний, системный таймер *SysTick*, отладочная система и карта памяти). Четырех гигабайтное адресное пространство *Cortex-M3* разделено на четко распределенные области кода программы, статического ОЗУ, устройств ввода-вывода и системных ресурсов. В отличие от ядра *ARM7, Cortex-M3* выполнено по Гарвардской архитектуре и, поэтому, имеет несколько шин, позволяющие выполнять операции параллельно. Семейство *Cortex* имеет возможность оперировать с фрагментированными данными (*unaligned data*), что также отличает его от предшествующих архитектур *ARM*. Этим гарантируется максимальная эффективность использования внутреннего статического ОЗУ. Семейство *Cortex* также поддерживает возможности установки и сброса бит в пределах двух областей памяти размером один Мбайт по методу *bit banding*. Этот метод предоставляет эффективный доступ к регистрам и флагам УВВ, расположенных в области статического ОЗУ, и исключает необходимость интеграции полнофункционального битового процессора. Под *ARM*-архитектуру существует довольно широкий выбор программных средств разработки. Приведем лишь основные и самые популярные программные пакеты на российском рынке. Встроенным контроллером оперативной памяти комплектуется любой процессор *ARM Cortex A7*. Характеристики технического плана указывают на то, что он ориентирован на работу в связке с ОЗУ стандарта LPDDR3. Рекомендованные частоты функционирования оперативной памяти в данном случае равны 1066 МГц или 1333 МГц. Максимальный же размер ОЗУ, который можно встретить на практике, для данной модели чипа равен 2 Гб.

Таблица 1 – Среды разработки ПО для *ARM* микроконтроллеров

Инструментарий	Среда разработки	*C/C++* компилятор	Ограничение Си-иструментария	Программатор-отладчик	Ссылка
IAR Systems	*Embedded Workbench*	*IAR C/C++*	32 Кбайт или полная с ограничением на 30 дней	*J-Link ST-Link*	*www.iar.com*
Keil	*uVision (MDK-ARM)*	*Keil C/C++*	32 Кбайт	*ULink ST-Link*	*www.keil.com*
Raisonance	*Ride7 + RKIT-ARM*	*GCC C/C++*	Нет, ограничения по отладке	*RLink*	*www.raisonance.com*
Atollic	*TrueStudio*	*GCC C/C++*	Нет, ограничения по функционалу	*ST-Link STICE*	*www.atollic.com*
Open source	*Eclipse*	*GCC C/C++*	Без ограничений	*ARM-Link*	*www.eclipse.org*

Наиболее популярными среди разработчиков для разработки ПО под ARM архитектуру являются инструментарии от компаний *Keil* и *IAR Systems*. Это обусловлено наиболее продвинутыми С-инструментариями с точки зрения оптимизации и компактности кода. Также, помимо лидирующих позиций в С-инструментариях, данные компании предоставляют широкие наборы дополнительного ПО – операционные системы реального времени, USB-стеки, TCP/IP-стеки и многое другое, но за дополнительную плату. Следует отметить популярность средств на основе компилятора *GCC*. Существуют как платные их варианты, так и бесплатные. Помимо всего, *GCC* является лидером по количеству поддерживаемых процессоров и операционных систем. На рынке

существует огромный выбор оценочных плат для *STM32* как от *STMicroelectronics*, так и от сторонних производителей. Для ознакомления и освоения *STM32* компания *STMicroelectronics* разработала линейку оценочных плат «*Discovery*»: для восьмибитных микроконтроллеров – *STM8S-Discovery* и *STM8L-Discovery*, для 32-битных – *STM32VLDiscovery*. Особенность данных оценочных плат заключается в завершенном решении для старта разработки программного обеспечения на микроконтроллерах – сам микроконтроллер с необходимой обвязкой и внешними компонентами, а также интегрированный программатор-отладчик *ST-Link*. Это полноценное решение, не требующее дополнительных затрат. Используя эти платы в собственных разработках, можно применять для программирования и отладки собственных приложений встроенный *ST-Link* через выведенный внешний разъем. На рисунке 1.10 изображена отладочная плата *STM32VLDiscovery*.

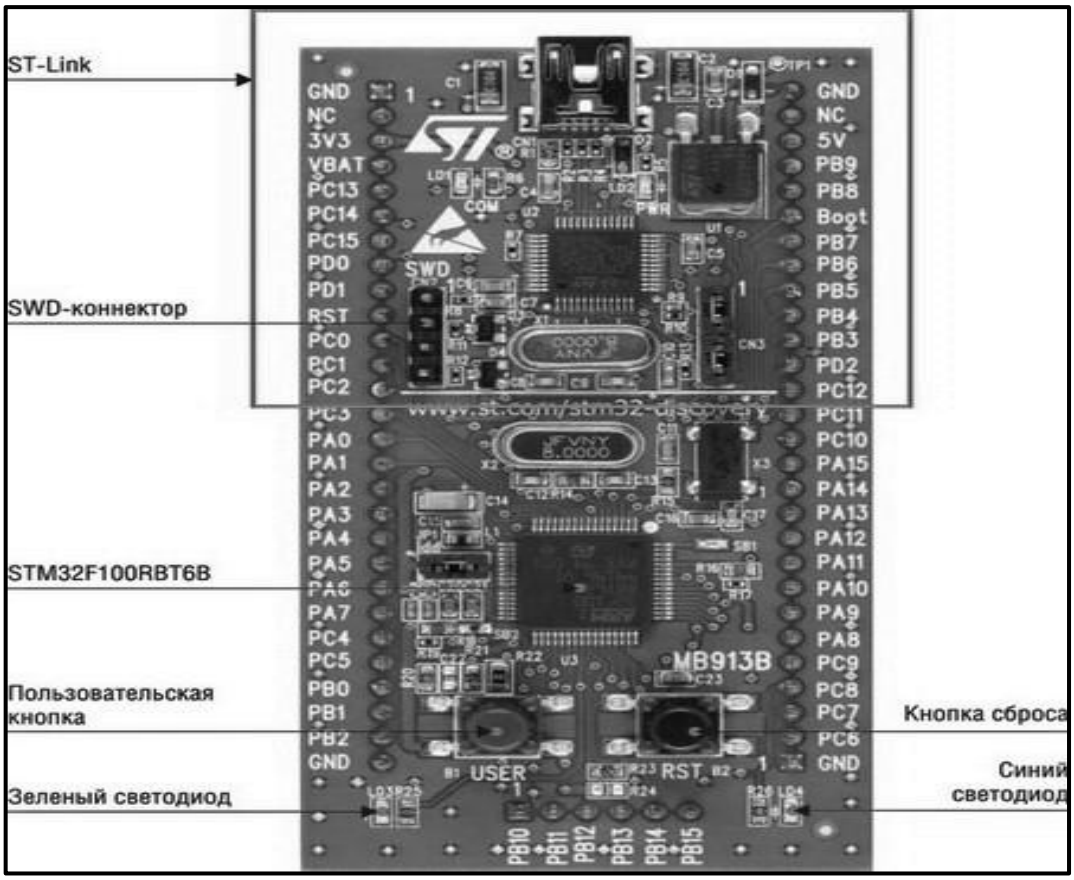


Рисунок 1.10 – Отладочная плата *STM32VLDiscovery*

В основе платы – микроконтроллер линейки *«Value Line»* *STM32F100RBT6*, программатор-отладчик *ST-Link* с выведенным разъемом SWD, механические кнопки, светодиоды и обвязка. Свободные ножки микроконтроллера выведены на внешние разъемы.

STMicroelectronics для облегчения труда разработчиков предоставляет бесплатные стандартные библиотеки периферии для своих микроконтроллеров и, в частности, для семейства *STM32*. На рисунке 1.11 представлена функциональная структура стандартной библиотеки периферии *STM32*.

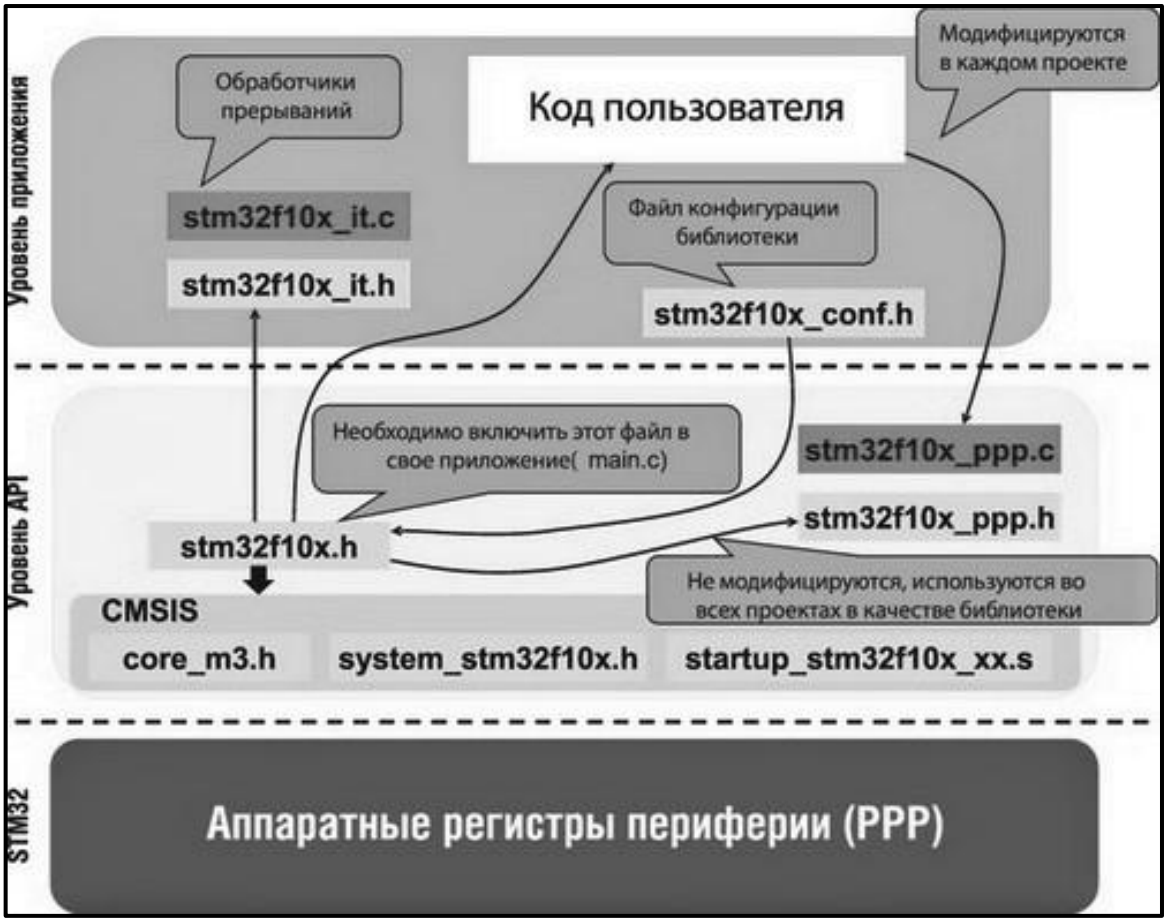


Рисунок 1.11 - Функциональная структура стандартной библиотеки периферии

Стандартная библиотека периферии написана в соответствии со стандартом *ANSI C* и может использоваться с любым компилятором. Структура

библиотеки не так сложна, как кажется на первый взгляд, и состоит из двух взаимодополняющих составляющих.

Первая составляющая – заголовочные файлы и файлы реализации всей периферии микроконтроллеров *STM32 – STM32F10x_StdPeriph_Driver*. Вся функциональность периферийных модулей описана в заголовочных файлах и файлах реализации. Например, для портов ввода-вывода это два файла – *stm32f10x_gpio.h и stm32f10x_gpio.c.*

Вторая составляющая – заголовочные файлы и файлы реализации самого ядра *ARM Cortex-M3* от компании *ARM – CMSIS (ARM® CortexTM Microcontroller Software Interface Standard*). *Я*дро *ARM Cortex-M3* выходит за рамки обычного понятия ядра микроконтроллера и представляет собой мини-микроконтроллер с периферией – встроенные системный таймер, контроллер прерываний и т.д. *CMSIS* предоставляет собой константы и определения, функции доступа к регистрам и периферийным модулям ядра, независимый интерфейс для операционных систем реального времени (RTOS). *CMSIS* состоит из трех файлов:

- *core_m3.h* – вспомогательные функции доступа к регистрам ядра;

- *startup_stm32f10x_xx.s*- набор файлов для каждой линейки семейства *STM32*, обеспечивающие инициализацию стека и таблицу векторов прерываний;

- *system_stm32f10x.h*- файл начальной инициализации тактовой частоты микроконтроллера.

Для использования стандартной библиотеки периферии необходимо в файл основной программы (обычно это *main.c*) включить файл *#include «stm32f10x.h»* и прописать определенные константы в свойствах проекта. Библиотекой предоставляются три файла, доступные для модификации пользователем – файл конфигурации библиотеки *stm32f10x_conf.h* и файлы прерываний *stm32f10x_it.h* и *stm32f10x_it.c.* Для использования определенных модулей периферии в проект необходимо добавить файлы реализации и сконфигурировать файл *stm32f10x_conf.h*. Под конфигурацией файла

stm32f10x_conf.h подразумевается раскомментирование строчек с названием периферийного модуля, предполагаемого для использования. Обработка прерываний происходит в заголовочном файле и файле реализации *stm32f10x_it.h u stm32f10x_it.c*. Функции обработчиков прерывания не должны содержать параметров – *void function(void).* Вся периферия описана в структурах данных языка Си, которые и используются для конфигурации периферийного модуля. Функции и константы для периферийных модулей начинаются с префиксов, совпадающих с именем периферийного модуля. Например, имена функций для портов ввода-вывода – *GPIO_Init(), GPIO_SetBits(), GPIO_ReadInputData()* и константы – *GPIOA, GPIO_Speed_50MHz, GPIO_Pin_0.*

PIC — серия микроконтроллеров, имеющих гарвардскую архитектуру и производимых американской компанией *Microchip Technology Inc*. Название PIC является сокращением от английского *peripheral interface controller*, что означает «контроллер интерфейса периферии»

PIC16F84 относится к семейству КМОП микроконтроллеров. Отличается тем, что имеет внутреннее 1024 x 14 бит *EEPROM* для программ, восьмибитовые данные и 64байт *EEPROM* памяти данных. При этом отличаются низкой стоимостью и высокой производительностью. Все команды состоят из одного слова (14 бит шириной) и исполняются за один цикл (одна Мкс при четырех МГц), кроме команд перехода, которые выполняются за два цикла (две мкс). *PIC16F84* имеет прерывание, срабатывающее от четырех источников, и восьмиуровневый аппаратный стек. Периферия включает в себя восьмибитный таймер/счетчик с восьмибитным программируемым предварительным делителем (фактически 16 - битный таймер) и 13 линий двунаправленного ввода/вывода. Высокая нагрузочная способность (25 мА максимальный входной ток, 20 мА максимальный выходной ток) линий ввода/вывода упрощают внешние драйверы и, тем самым, уменьшается общая стоимость системы. Разработки на базе контроллеров *PIC16F84* поддерживается ассемблером, программным симулятором, внутрисхемным

эмулятором (только фирмы *Microchip*) и программатором. Встроенный автомат программирования *EEPROM* кристалла *PIC16F84* позволяет легко подстраивать программу и данные под конкретные требования даже после завершения ассемблирования и тестирования. Эта возможность может быть использована как для тиражирования, так и для занесения калибровочных данных уже после окончательного тестирования.

Обзор характеристик *PIC16F84*:

Высокоскоростной *RISC* процессор:

- только 35 простых команд;
- все команды выполняются за один цикл (одна Мкс), кроме команд перехода, выполняющихся за два цикла;
- рабочая частота 0 Гц - 4 МГц;
- 14- битовые команды;
- восьми- битовые данные;
- 1024 x 14 электрически перепрограммируемой программной памяти на кристалле (*EEPROM*);
- 36 x 8 регистров общего использования;
- 15 специальных аппаратных регистров *SFR*;
- 64 x 8 электрически перепрограммируемой *EEPROM* памяти для данных;
- восьмиуровневый аппаратный стек;
- прямая, косвенная и относительная адресация данных и команд.

Четыре источника прерывания:

- внешний вход *INT*;
- переполнение таймера *TMR*0;
- прерывание при изменении сигналов на линиях порта «B» (восьмиразрядный порт ввода/вывода);
- по завершению записи данных в память *EEPROM*.

Периферия и Ввод/Вывод:

- 13 линий ввода-вывода с индивидуальной настройкой;

- входной/выходной ток для управления светодиодами;

- максимальный входной ток - 20 мА;

- максимальный выходной ток - 25 мА;

- *TMR0*: восьми-битный таймер/счетчик *TMR0* с восьмибитным программируемым предварительным делителем.

Специальные свойства:

- автоматический сброс при включении;

- таймер включения при сбросе;

- таймер запуска генератора;

- *WatchDog* таймер (*WDT*) с собственным встроенным генератором, обеспечивающим повышенную надежность;

- *EEPROM* бит секретности для защиты кода;

- экономичный режим *SLEEP*;

- встроенное устройство программирования *EEPROM* памяти программ и данных используются только две ножки. КМОП технология;

- экономичная высокоскоростная КМОП *EPROM* технология;

- статический принцип в архитектуре.

Широкий диапазон напряжений питания и температур:

- коммерческий: от двух до шести В, от нуля до 70 °С;

- промышленный: от двух до шести В, от минус 40 до 70 °С;

- автомобильный: от двух до шести В, от 40 до 125 °С.

Низкое потребление:

- два мА типично для пяти В, четырех МГц;

- 15 мкА типично для двух В, 32 КГц;

- один мкА типично для *SLEEP* режима при двух В.

На рисунке 1.12 представлена обобщенная функциональная схема микроконтроллеров *PIC24F*.

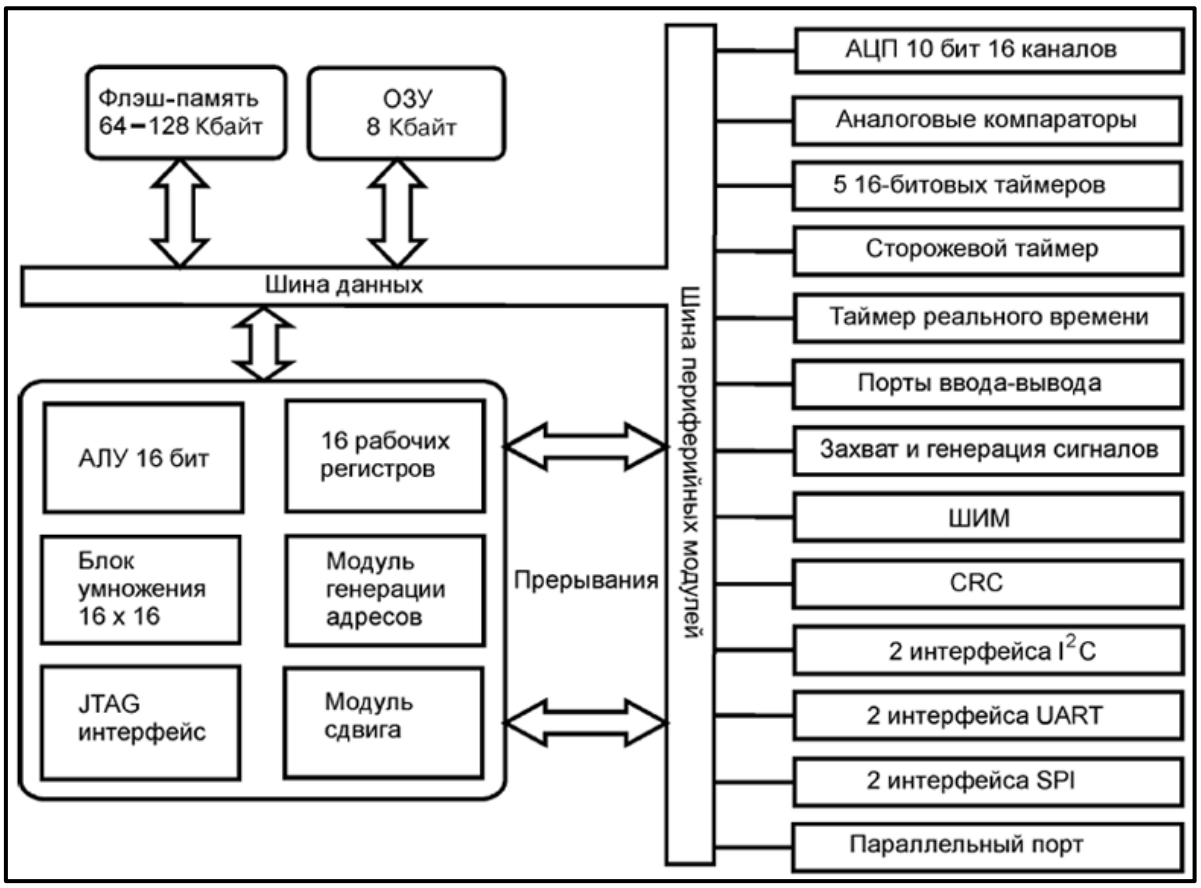


Рисунок 1.12 – Обобщенная функциональная схема микроконтроллеров *PIC24F*

Микроконтроллеры *PIC24F* взаимодействуют с различными внешними периферийными устройствами посредством интерфейсов I2C, SPI и UART. Для этого в состав устройства включены соответствующие модули, которые настраиваются и работают независимо друг от друга. Функциональность подсистемы асинхронной последовательной передачи данных улучшена за счет включения в UART аппаратно-программного модуля IrDA. Существенно улучшает рабочие характеристики модулей интерфейсов SPI и UART сохранение данных в буфере *FIFO*, который позволяет снизить непроизводительные траты процессорного времени на обработку передачи данных. В отличие от многих устройств семейства *PIC18*, в которых реализован порт параллельной передачи данных, работающий только в режиме «ведомого», в микроконтроллерах *PIC24F* имеется модуль параллельного

обмена данными, который позволяет работать как в режиме «ведомого», так и в режиме «ведущего». Это существенно расширяет возможности *PIC24F* при обмене данными с периферийными устройствами, имеющими параллельный интерфейс. Все устройства семейства *PIC24F* имеют один и тот же набор базовых периферийных модулей и отличаются объемом флэш-памяти. В микроконтроллерах семейства *Microchip* используется 10-битный аналого-цифровой преобразователь последовательного приближения. Аналого-цифровой преобразователь может работать в режиме автоматического сканирования входов и поддерживает различные режимы синхронизации. Модуль АЦП допускает автономную работу при переходе процессора в «спящий» режим или режим «холостого хода». Аналого-цифровой преобразователь может производить несколько последовательных выборок, накапливая результат в 16-уровневом буфере данных, и сохранять результат в одном из четырех форматов. Модуль аналоговых компараторов включает в себя два компаратора, которые используются при реализации широкого класса функциональных узлов, например детектора перехода через ноль в схеме синхронизации по переменному току 50 Гц, или при создании более сложных устройств, таких, как 16-битный сигма-дельта аналого-цифровой преобразователь. Микроконтроллеры *PIC24F* включают пять модулей таймеров общего назначения разрядностью 16 бит. Все пять таймеров обладают общими базовыми функциональными возможностями. Регистры периода всех таймеров могут использоваться для генерации прерывания при совпадении содержимого такого регистра с текущим содержимым регистра таймера. Во всех таймерах предусмотрены режим запуска/останова по внешнему сигналу и генерация прерывания по спаду внешнего сигнала. Четыре из пяти таймеров могут объединяться попарно для формирования 32-битных таймеров. С модулями таймеров тесно связан пятиканальный модуль захвата входных сигналов и пятиканальный модуль генерации цифровых сигналов. Модуль захвата входных сигналов используется для измерения интервалов между событиями. Минимальная разрешающая способность при таком измерении равна

длительности одного машинного цикла. Для синхронизации временных меток модуль захвата входных сигналов использует в качестве базовых второй и третий Таймеры. Модуль генерации цифровых сигналов используется для генерации одиночных импульсов и импульсных последовательностей и имеет пять каналов. Для отсчета интервалов времени при формировании таких сигналов модуль генерации цифровых сигналов использует в качестве базовых таймер два и таймер три. Выходные сигналы этого модуля могут использоваться для управления обычными (термоэлементы) и индуктивными (электродвигатели) нагрузками, а также для синтеза голосовых сигналов. В микроконтроллерах семейства *PIC24F* имеется два модуля универсального асинхронного приемопередатчика (UART), которые позволяют реализовать обмен данными в соответствии со стандартами *RS-232* и *RS-485*. Приемопередатчик обеспечивает обмен восьми или девятибитными данными с контролем четности или без такового с одним или двумя стоповыми битами. Устройство поддерживает функцию аппаратного контроля обмена данными, что обеспечивает высокий уровень надежности и производительности, позволяя подключать к микроконтроллеру различные периферийные устройства, например модемы. Скорость передачи данных может изменяться от 15 Бод до 100000 Бод. Повышение производительности операций обмена данными и разгрузка процессора обеспечиваются за счет использования четырехуровневых буферов входных/выходных данных и прерываний. Кроме того, модуль UART поддерживает передачу информации по протоколам *IrDA* и *LIN*. Для обмена данными между различными устройствам в настоящее время очень часто используются протоколы *I2C* и *SPI*. Микроконтроллеры *PIC24F* имеют интегрированные периферийные модули для поддержки этих протоколов. В модуле *I2C* применяется аппаратно-программный алгоритм, позволяющий выбрать режим «ведущего» или «ведомого». Модуль поддерживает как семи, так и 10-битную адресацию устройств на шине *I2C*, при этом тактовая частота шины *I2C* может быть задана равной 100 кГц или 400 кГц. Модули *SPI* микроконтроллеров *PIC24F* можно сконфигурировать для работы с двумя,

тремя или четырьмя сигнальными линиями. В режиме с двумя линиями (синхронизации и данных) интерфейс *SPI* можно использовать для приема сигналов датчиков. Для работы с аналого-цифровыми преобразователями, сдвиговыми регистрами и микросхемами памяти *EEPROM* используется, как правило, трехпроводной интерфейс. Еще один периферийный модуль — модуль часов реального времени с календарем (*RTCC*) — предназначен для точного отсчета времени в течение длительных интервалов и оперирует с датами и временем. Он может выполнять функцию будильника, включая внешнее устройство в определенный момент времени в будущем. Модуль *RTCC* синхронизируется непосредственно от внешнего источника тактового сигнала частотой 32 кГц. Из этого сигнала посредством предделителя формируется внутренний сигнал с периодом 0,5 с, который используется для синхронизации регистров модуля, содержащих элементы даты (год, месяц, день, день недели, часы, минуты и секунды). Данные хранятся в регистрах в BCD-формате. Функция будильника может быть запрограммирована на определенный месяц, день, день недели, час, минуту и секунду. Кроме того, поскольку регистры модуля часов реального времени работают на очень низкой частоте, это позволяет минимизировать энергопотребление устройства. Часы реального времени могут функционировать и в том случае, если процессор находится в «спящем» режиме. Для управления внешними устройствами можно использовать выходной сигнал модуля частотой один Гц или сигнал, формируемый при срабатывании будильника. Модуль параллельного порта позволяет легко реализовывать аппаратно-программный восьмибитный интерфейс с внешними устройствами и модулями памяти. Модуль поддерживает мультиплексирование шин адреса/данных, позволяя передавать по восьмибитной шине 16-битные данные. В модуле предусмотрена работа с 16 адресными линиями, что дает возможность адресовать до 64 Кбайт памяти, а при использовании дополнительных линий адреса, в качестве которых могут быть задействованы выводы портов общего назначения, — и большее пространство адресов. В модуле предусмотрена функция

автоинкремента/автодекремента адреса, что позволяет оптимизировать передачу больших блоков данных. Модуль контроля достоверности данных с использованием циклического избыточного кода (*CRC*). Этот модуль находит применение при контроле ошибок обмена данными с периферийным оборудованием и памятью, особенно при работе с коммуникационным оборудованием по протоколам CAN, USB и Ethernet. Микроконтроллеры семейства *PIC24F* имеют 16-разрядную шину данных, что существенно повышает их функциональные возможности по сравнению с устройствами семейства *PIC18*. Длина инструкции (команды) процессора в микроконтроллерах серии *PIC24* равна 24 битам, а частота выполнения команд процессора в два раза меньше, чем частота синхронизации устройства.

Если использовать принятые в документации фирмы *Microchip* обозначения, то это соотношение можно выразить формулой:

$$FCY = FOSC/2, \quad (1)$$

где FOSC — тактовая частота синхронизации микроконтроллера, а FCY — тактовая частота процессора. Счетчик команд (*Program Counter, PC*) имеет разрядность 23 бита и позволяет адресовать до четырех миллионов инструкций в памяти программ. Высокая производительность микроконтроллера обеспечивается за счет использования механизма конвейеризации инструкций, при котором выборка и декодирование следующей инструкции процессора осуществляются на этапе выполнения предыдущей. При этом все инструкции выполняются за один машинный цикл, за исключением инструкций передачи управления, инструкций, оперирующих двойными словами, и инструкций табличного чтения/записи. Кроме того, определенную оптимизацию загрузки процессора можно осуществить за счет применения инструкции цикла *repeat*.

Микроконтроллеры семейства *PIC24F* имеют 16 рабочих регистров, оперирующих данными размером в одно слово (16 бит). Каждый из этих регистров может выступать в роли регистра данных, адреса или смещения.

Шестнадцатый по счету регистр (W15) используется в качестве указателя стека (*SP, Stack Pointer*) в операциях вызова процедур и обработчиков прерываний. В микроконтроллерах PIC24F предусмотрена возможность отображения верхних 32 Кбайт памяти данных на адресное пространство памяти программ с выравниванием по границе слова, для чего используется восьмибитный регистр *PSVPAG (Program Space Visibility Page).* Разработчики семейства *PIC24F* предусмотрели обратную совместимость инструкций и режимов адресации процессора с микроконтроллерами семейства *PIC18* за счет прямого включения подмножества инструкций *PIC18* в систему команд *PIC24*, а также посредством использования макросов. На рисунке 1.13 представлена схема организации памяти программ микроконтроллеров *PIC24FJ128GA.*

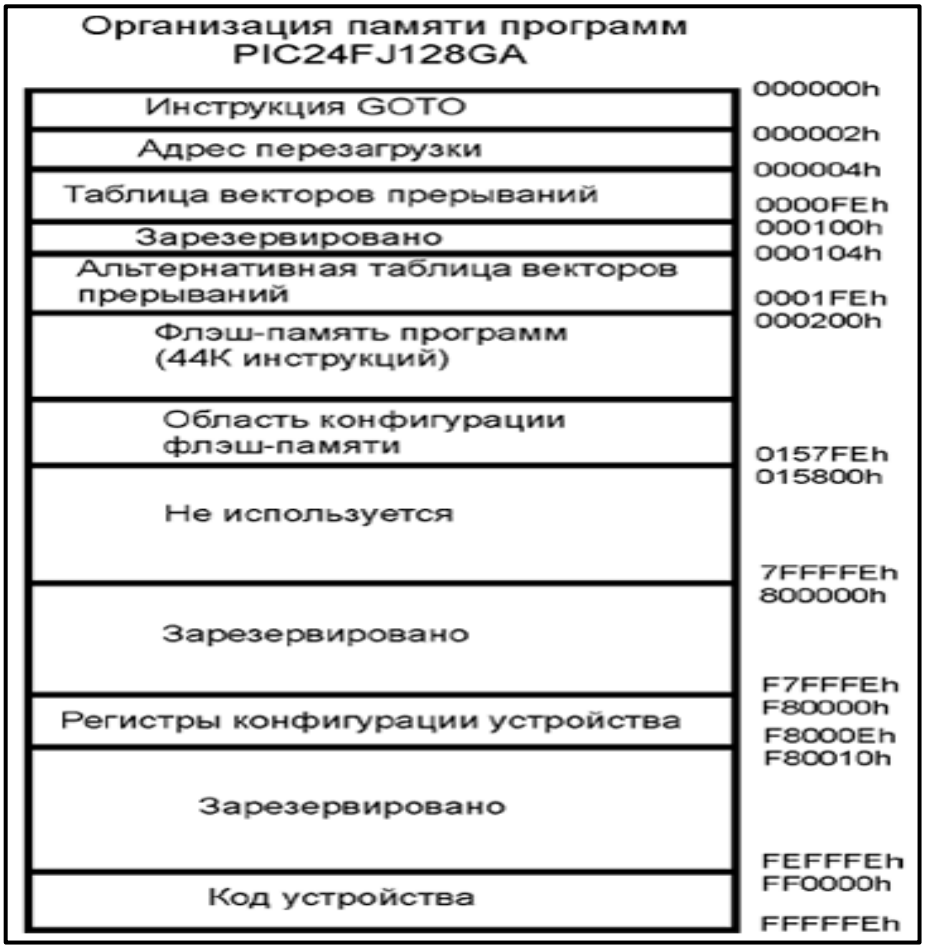


Рисунок 1.13 - Схема организации памяти программ микроконтроллеров *PIC24FJ128GA*

В микроконтроллерах семейства *PIC24F* реализована гарвардская архитектура, в которой память программ и память данных разделены, что позволяет осуществлять прямой доступ к памяти программ из памяти данных во время выполнения программного кода. На рисунке 1.14 представлена схема распределения памяти данных *PIC24FJ128GA*.

Рисунок 1.14 - Схема распределения памяти данных

Пользовательским программам доступна область памяти в диапазоне адресов 0x000000-0x7FFFFF, за исключением области конфигурации устройства, доступ к которой осуществляется посредством инструкций *tblrd/tblwt*. Память программ организована в виде блоков, адресуемых пословно. Хотя адрес памяти является 24-битным, более удобно рассматривать любой адрес в виде младшего и старшего слова, при этом старший байт старшего слова не используется и равен нулю. Младшее слово всегда располагается по четному адресу, а старшее — по нечетному. Адреса памяти программ всегда выравниваются по границе слова и при выполнении программного кода всегда инкрементируются и декрементируются на два. Область памяти программ между адресами 0x000000 и 0x000200 зарезервирована для векторов прерываний. По адресам 0x000000 и 0x000002 размещается команда перехода к фактическому началу программы, при этом по первому адресу располагается код операции инструкции goto, а по второму — собственно адрес точки входа в программу. Также в памяти программ размещаются две таблицы векторов прерываний. Одна из них располагается в диапазоне адресов 0x000004-0x0000FF, а вторая (альтернативная) — в диапазоне адресов 0x000100-0x0001FF. Адресное пространство памяти данных микроконтроллеров *PIC24F* представляет собой непрерывную область 16-битных адресов с линейной адресацией. Адрес для доступа к данным формируется двумя модулями генерации адресов, один из которых используется в операциях записи, а другой в операциях чтения данных. Все действующие адреса памяти данных (*Effective Addresses, EA*) имеют размер 16 бит, что позволяет адресовать 64 Кбайт памяти. Нижняя половина адресов памяти данных, для которых старший бит действующего адреса равен нулю, используется для адресации данных, а старшая половина адресов, для которых старший бит адреса равен единице, — для отображения памяти программ на память данных (*Program Space Visibility Area, PSVA*). Микроконтроллеры *PIC24FJ128GA010* имеют объем памяти данных восемь Кбайт; при обращении к памяти данных по адресу, лежащему за пределами этой области,

возвращается нулевой байт или слово. Данные в памяти выравниваются по границе слова, при этом младшие байты имеют четные, а старшие — нечетные адреса. Для обратной совместимости с восьмибитными микроконтроллерами в систему команд *PIC24F* включены команды, поддерживающие операции как со словами, так и с отдельными байтами. При этом процессор распознает, когда выполняется операция над байтом, а когда – над словом, и генерирует соответствующий адрес. Например, если в команде используется косвенная адресация с постинкрементом, такая, как [W1++], то для байтовой команды генерируется адрес, равный W1+1, а для команды, работающей со словом, — W1+2. При любых операциях со словами данные должны иметь четный адрес, поэтому компилятор будет выдавать ошибку при сборке программ, если в программном коде происходит обращение к слову по нечетному адресу. Если такая ситуация возникает при выполнении программы, то генерируется ошибка адресации и вызывается обработчик этой исключительной ситуации. По этой причине следует соблюдать осторожность, если в программе одновременно присутствуют инструкции, работающие с байтами и со словами. Для повышения эффективности работы программ в систему команд включен ряд инструкций, позволяющих преобразовывать байты данных в слова. Это инструкция расширения знака *se* (*Sign-Extend*), приводящая восьмибитное число со знаком к 16-битному формату, и инструкция расширения нуля *ze (Zero-Extend)*, заполняющая старший байт беззнакового целого нулями. Большинство инструкций процессора могут работать как с байтами, так и со словами, но некоторые инструкции оперируют только со словами. Первые два килобайта адресного пространства памяти данных (от 0x0000 до 0x7FF) выделяются для регистров специальных функций (*Special Function Register, SFR),* которые используются ядром и периферийными модулями микроконтроллера *PIC24F* для выполнения различных операций. Эти регистры распределены между модулями, с которыми они связаны, и, как правило, группируются по принадлежности к тому или иному периферийному модулю. Большинство адресов, выделяемых под регистры специальных функций, на

данный момент не задействованы и читаются как нуль. Инструкции процессора в *PIC24F* могут быть регистровыми и адресными. Основное различие между этими типами инструкций состоит в том, что регистровые инструкции модифицируют содержимое регистров, в то время как адресные инструкции используют содержимое рабочих регистров для доступа к данным, находящимся в памяти. В свою очередь адресные инструкции могут быть как с прямой регистровой адресацией *(register direct addressing),* так и с косвенной регистровой адресацией *(register indirect addressing).* Прямая регистровая адресация используется для доступа к содержимому 16 рабочих регистров W0-W15, причем обращение возможно как к целому слову (16 бит), так и к байту.

Систему инструкций (команд) микроконтроллеров *PIC24F* можно разделить на несколько групп: команды перемещения, команды математических операций, команды логических операций, команды сдвига и циклического сдвига, команды работы с битами; команды сравнения/выбора, команды условных переходов, команды работы со стеком, команды управления. Косвенная регистровая адресация применяется для доступа к любой ячейке памяти данных посредством формирования исполнительного адреса данных по содержимому одного или нескольких рабочих регистров. Таким образом, содержимое рабочего регистра или регистров является указателем на область памяти данных. Дополнительные возможности такого метода адресации обеспечиваются механизмами прединкремента и постинкремента содержимого регистра, что позволяет последовательно обрабатывать непрерывный диапазон адресов в памяти данных. Адресация со смещениемпредполагает, что адресная часть команды включает в себя как минимум одно поле. В нем содержится константа, смысл которой в разных вариантах адресации со смешением может меняться. Константа может представлять собой некий базовым адрес, к которому добавляется хранящееся в регистре смещение. Методы формирования эффективного адреса *(EA)* с помощью косвенной регистровой адресации показаны в Таблице 2.

Таблица 2 - Формирование эффективного адреса методом косвенной регистровой адресации

Режим адресации	Байтовая адресация	Адресация слова	Описание
Косвенная без модификации	EA = [Wn]	EA = [Wn]	Эффективный адрес формируется по содержимому регистра Wn
Косвенная с постинкрементом	EA = [Wn]+=1	EA = [Wn]+=2	Эффективный адрес формируется по содержимому регистра Wn, затем содержимое Wn инкрементируется
Косвенная с постдекрементом	EA = [Wn]–=1	EA = [Wn]–=2	Перед формированием эффективного адреса содержимое Wn инкрементируется
Косвенная с прединкрементом	EA = [Wn+=1]	EA = [Wn+=2]	Перед формированием эффективного адреса содержимое Wn
Косвенная с преддекрементом	EA = [Wn–=1]	EA = [Wn–=2]	Перед формированием эффективного адреса содержимое Wn декрементируется
Косвенная со смещением, заданным в регистре	EA = [Wn + Wb]	EA = [Wn + Wb]	Эффективный адрес формируется как сумма Wn и Wb
Косвенная со смещением, заданным константой	EA = [Wn + Slit5]	EA = [Wn + 2*Slit5]	Эффективный адрес формируется как сумма Wn и константы Slit5 (значение в диапазоне от –16 до +15)

Команды условных/безусловных переходов предназначены для управления ходом вычислительного процесса и осуществляют переход к

различным секциям программы по результатам проверки флагов состояния процессора. В предыдущих примерах программного кода мы уже использовали некоторые из этих команд, сейчас же остановимся на них более подробно. Наиболее широко при разработке программного обеспечения используются инструкции *bra xx* (xx — код условия), *call* и *goto*. Команда *bra xx* осуществляет переход по адресу программы, содержащемуся в теле инструкции, при выполнении условия xx. В общей сложности эта команда может проверять выполнение следующих условий:

- C — установлен флаг переноса;
- GE — больше или равно;
- GEU — больше или равно для операций без знака;
- GT — больше;
- GTU — больше для операций без знака;
- LE — меньше или равно;
- LEU — меньше или равно для операций без знака;
- LT — меньше;
- LTU — меньше для операций без знака;
- N — результат предыдущей операции отрицательный;
- NC — перенос (заем) отсутствует;
- NN — результат предыдущей операции неотрицательный;
- NOV — переполнения нет;
- NZ — результат предыдущей операции ненулевой;
- OV — возникло переполнение;
- Z — установлен бит нуля в регистре состояния.

Например, команда *bra z, label1* выполнит переход на метку *label1* программы, если установлен бит Z в регистре состояния процессора. Кроме того, команда *bra* имеет две модификации, позволяющие выполнять безусловный переход:

- *bra iaoea*, где метка должна находиться в пределах от –32768 до +32767 слов памяти программ относительно команды bra;

- *bra Wn*, где Wn — рабочий регистр.

К командам переходов относятся и команды вызова процедур *call* и безусловного перехода *goto*. Инструкции *call* и *goto* в качестве операнда могут использовать либо метку, либо один из рабочих регистров. Если в качестве адреса перехода использовать содержимое какого-либо из рабочих регистров, то с помощью одной инструкции *bra* или *call* можно организовать несколько ветвлений в программе, например, по принципу оператора *switch…case* языка Си.

Микроконтроллеры серии *AVR* являются быстрыми. Большинство инструкций процессор микроконтроллера выполняет за один цикл. Микроконтроллеры *AVR* примерно в четыре раза быстрее, чем *PIC*. Кроме того, они потребляют немного энергии и могут работать в четырех режимах экономии энергии. Большинство контроллеров *AVR* являются восьмиразрядными, хотя сейчас существует и 32-разрядная разновидность контроллеров *AVR32*. Кроме того, *AVR* принадлежат к типу RISC-микроконтроллеров. Архитектура RISC (*Complex Instruction Set Computers*) означает, что набор инструкций, которые может выполнять процессор устройства, является ограниченным, но, в то же время, подобная архитектура дает преимущество в скорости. Противоположностью архитектуры RISC является архитектура CISC (*Complex Instruction Set Computers*). Всего контроллер *AVR* имеет 32 восьмибитных регистра общего назначения. В течение цикла процессор берет данные из двух регистров и помещает их в арифметико-логическое устройство (АЛУ), которое производит операцию над данными и помещает их в произвольный регистр. АЛУ может выполнять как арифметические, так и логические действия над операндами. Также АЛУ может выполнять и действия с одним операндом. При этом контроллер не имеет регистра-аккумулятора, в отличие от контроллеров семейства *i8051* – для операций могут использоваться любые регистры, и результат операции также может быть помещен в любой регистр. Контроллер соответствует Гарвардской вычислительной архитектуре, согласно которой компьютер имеет отдельную

память для программ и данных. Поэтому в то время, пока выполняется одна инструкция, происходит предварительное извлечение из памяти следующей инструкции. Котроллер способен выполнять одну инструкцию за цикл. Отсюда следует, что если тактовая частота контроллера составляет один МГц, то его производительность составит один млн операций в секунду. Чем выше тактовая частота контроллера, тем выше будет его скорость. Однако при выборе тактовой частоты контроллера следует соблюдать разумный компромисс между его скоростью и энергопотреблением. Помимо флэш-памяти и процессора контроллер имеет такие устройства, как порты ввода-вывода, аналого-цифровой преобразователь, таймеры, коммуникационные интерфейсы – *I2C, SPI* и последовательный порт UART. Все эти устройства могут контролироваться программно. Сердцем микроконтроллеров *AVR* является восьмибитное микропроцессорное ядро или центральное процессорное устройство (ЦПУ), построенное на принципах RISC-архитектуры. Основой этого блока служит арифметико-логическое устройство (АЛУ). По системному тактовому сигналу из памяти программ в соответствии с содержимым счетчика команд (*Program Counter - PC*) выбирается очередная команда и выполняется АЛУ. Во время выбора команды из памяти программ происходит выполнение предыдущей выбранной команды, что и позволяет достичь быстродействия один MIPS на один МГц. АЛУ подключено к регистрам общего назначения *РОН (General Purpose Registers - GPR)*. Регистров общего назначения всего 32, они имеют байтовый формат, то есть каждый из них состоит из восьми бит. *РОН* находятся в начале адресного пространства оперативной памяти, но физически не являются ее частью. Поэтому к ним можно обращаться двумя способами (как к регистрам и как к памяти). Такое решение является особенностью *AVR* и повышает эффективность работы и производительность микроконтроллера. Отличие между регистрами и оперативной памятью состоит в том, что с регистрами можно производить любые операции (арифметические, логические, битовые), а в оперативную память можно лишь записывать данные из регистров. В микроконтроллерах *AVR* реализована Гарвардская архитектура,

в соответствии с которой разделены не только адресные пространства памяти программ и памяти данных, но и шины доступа к ним. Каждая из областей памяти данных (оперативная память и *EEPROM*) также расположена в своем адресном пространстве. Память программ предназначена для хранения последовательности команд, управляющих функционированием микроконтроллера, имеет 16-ти битную организацию. Все *AVR* имеют Flash-память программ, которая может быть различного размера - от одного Кбайта до 256 КБайт. Ее главное достоинство в том, что она построена на принципе электрической перепрограммируемости, т. е. допускает многократное стирание и запись информации. Программа заносится во Flash-память *AVR* как с помощью обычного программатора, так и с помощью *SPI*-интерфейса, в том числе непосредственно на собранной плате. Возможностью внутрисхемного программирования (функция *ISP*) через коммуникационный интерфейс *SPI* обладают все микроконтроллеры *AVR*, кроме *Tiny11* и *Tiny28*. Все микроконтроллеры семейства *Mega* имеют возможность самопрограммирования, т. е. самостоятельного изменения содержимого своей памяти программ. Эта особенность позволяет создавать на их основе очень гибкие системы, алгоритм работы которых будет меняться самим микроконтроллером в зависимости от каких-либо внутренних условий или внешних событий. Гарантированное число циклов перезаписи Flash-памяти у микроконтроллеров *AVR* второго поколения составляет не менее 10 тыс. циклов при типовом значении 100 тыс. циклов. Память данных разделена на три части: регистровая память, оперативная память (ОЗУ - оперативное запоминающее устройство или *RAM*) и энергонезависимая память (ЭСППЗУ или *EEPROM*). Регистровая память включает 32 регистра общего назначения (*РОН* или *GPR*), объединенных в файл, и служебные регистры ввода/вывода (РВВ). И те и другие расположены в адресном пространстве ОЗУ, но не являются его частью. В области регистров ввода/вывода расположены различные служебные регистры (регистры управления микроконтроллером, регистры состояния и т. п.), а также регистры управления периферийными устройствами, входящими в

состав микроконтроллера. Для долговременного хранения различной информации, которая может изменяться в процессе функционирования микроконтроллерной системы, используется EEPROM-память. Все *AVR* имеют блок энергонезависимой электрически перезаписываемой памяти данных *EEPROM* от 64 Байт до четырех КБайт. Этот тип памяти, доступный программе микроконтроллера непосредственно в ходе ее выполнения, удобен для хранения промежуточных данных, различных констант, коэффициентов, серийных номеров, ключей и т.п. *EEPROM* может быть загружена извне как через *SPI* интерфейс, так и с помощью обычного программатора. Число циклов стирание/запись - не менее 100 тыс. Внутренняя оперативная статическая память *Static RAM (SRAM)* имеет байтовый формат и используется для оперативного хранения данных. Размер оперативной памяти может варьироваться у различных чипов от 64 Байт до четырех КБайт. Число циклов чтения и записи в *RAM* не ограничено, но при отключении питающего напряжения вся информация теряется. Периферия микроконтроллеров *AVR* включает: порты (от трех до 48 линий ввода и вывода), поддержку внешних прерываний, таймеры-счетчики, сторожевой таймер, аналоговые компараторы, 10-разрядный восьмиканальный АЦП, интерфейсы *UART, JTAG* и *SPI*, устройство сброса по понижению питания, широтно-импульсные модуляторы.

Порты ввода/вывода *AVR* имеют число независимых линий "вход/выход" от трех до 53. Каждая линия порта может быть запрограммирована на вход или на выход. Мощные выходные драйверы обеспечивают токовую нагрузочную способность 20 мА на линию порта (втекающий ток) при максимальном значении 40 мА, что позволяет, например, непосредственно подключать к микроконтроллеру светодиоды и биполярные транзисторы. Общая токовая нагрузка на все линии одного порта не должна превышать 80 мА (все значения приведены для напряжения питания пять В). Архитектурная особенность построения портов ввода/вывода у *AVR* заключается в том, что для каждого физического вывода существует три бита контроля/управления, а не два, как у распространенных восьмиразрядных микроконтроллеров *(Intel, Microchip,*

Motorola и т.д.). Это позволяет избежать необходимости иметь копию содержимого порта в памяти для безопасности и повышает скорость работы микроконтроллера при работе с внешними устройствами, особенно в условиях внешних электрических помех. Система прерываний - одна из важнейших частей микроконтроллера. Все микроконтроллеры *AVR* имеют многоуровневую систему прерываний. Прерывание прекращает нормальный ход программы для выполнения приоритетной задачи, определяемой внутренним или внешним событием. Для каждого такого события разрабатывается отдельная программа, которую называют подпрограммой обработки запроса на прерывание, и размещается в памяти программ. При возникновении события, вызывающего прерывание, микроконтроллер сохраняет содержимое счетчика команд, прерывает выполнение центральным процессором текущей программы и переходит к выполнению подпрограммы обработки прерывания. После выполнения подпрограммы прерывания осуществляется восстановление предварительно сохраненного счетчика команд и процессор возвращается к выполнению прерванной программы. Для каждого события может быть установлен приоритет. Понятие приоритет означает, что выполняемая подпрограмма прерывания может быть прервана другим событием только при условии, что оно имеет более высокий приоритет, чем текущее. В противном случае центральный процессор перейдет к обработке нового события только после окончания обработки предыдущего. Микроконтроллеры *AVR* имеют в своем составе от одного до четырех таймеров/счетчиков с разрядностью восемь или 16 бит, которые могут работать и как таймеры от внутреннего источника тактовой частоты, и как счетчики внешних событий. Их можно использовать для точного формирования временных интервалов, подсчета импульсов на выводах микроконтроллера, формирования последовательности импульсов, тактирования приемопередатчика последовательного канала связи. В режиме ШИМ *(PWM)* таймер/счетчик может представлять собой широтно-импульсный модулятор и используется для генерирования сигнала с программируемыми частотой и скважностью. Таймеры/счетчики способны вырабатывать запросы

прерываний, переключая процессор на их обслуживание по событиям и освобождая его от необходимости периодического опроса состояния таймеров. Поскольку основное применение микроконтроллеры находят в системах реального времени, таймеры/счетчики являются одним из наиболее важных элементов. Сторожевой таймер *(WatchDog Timer)* предназначен для предотвращения катастрофических последствий от случайных сбоев программы. Он имеет свой собственный RC-генератор, работающий на частоте один МГц. Как и для основного внутреннего RC-генератора, значение один МГц является приближенным и зависит прежде всего от величины напряжения питания микроконтроллера и от температуры. Идея использования сторожевого таймера предельно проста и состоит в регулярном его сбрасывании под управлением программы или внешнего воздействия до того, как закончится его выдержка времени и не произойдет сброс процессора. Если программа работает правильно, то команда сброса сторожевого таймера должна регулярно выполняться, предохраняя поцессор от сброса. Если же микропроцессор случайно вышел за пределы программы (например, от сильной помехи по цепи питания) либо зациклился на каком-либо участке программы, команда сброса сторожевого таймера скорее всего не будет выполнена в течение достаточного времени и произойдет полный сброс процессора, инициализирующий все регистры и приводящий систему в рабочее состояние. Аналоговый компаратор *(Analog Comparator)* сравнивает напряжения на двух выводах микроконтроллера. Результатом сравнения будет логическое значение, которое может быть прочитано из программы. Выход аналогового компаратора можно включить на прерывание от аналогового компаратора. Пользователь может установить срабатывание прерывания по нарастающему или спадающему фронту или по переключению. Аналого-цифровой преобразователь (АЦП) служит для получения числового значения напряжения, поданного на его вход. Этот результат сохраняется в регистре данных АЦП. Какой из выводов (пинов) микроконтроллера будет являться входом АЦП, определяется числом, занесенным в соответствующий регистр. Универсальный асинхронный или

универсальный синхронно/асинхронный приемопередатчик *(Universal Synchronous/Asynchronous Receiver and Transmitter - UART* или *USART)* - удобный и простой последовательный интерфейс для организации информационного канала обмена микроконтроллера с внешним миром. Способен работать в дуплексном режиме (одновременная передача и прием данных). Он поддерживает протокол стандарта *RS-232*, что обеспечивает возможность организации связи с персональным компьютером. (Для стыковки МК и компьютера обязательно понадобится схема сопряжения уровней сигналов. Для этого существуют специальные микросхемы, например *MAX232*.)
Последовательный периферийный трехпроводный интерфейс *SPI (Serial Peripheral Interface)* предназначен для организации обмена данными между двумя устройствами. С его помощью может осуществляться обмен данными между микроконтроллером и различными устройствами, такими, как цифровые потенциометры, ЦАП/АЦП, FLASH-ПЗУ и др. С помощью этого интерфейса удобно производить обмен данными между несколькими микроконтроллерами *AVR*. Кроме того, через интерфейс SPI может осуществляться программирование микроконтроллера. Двухпроводной последовательный интерфейс TWI *(Two-wire Serial Interface)* является полным аналогом базовой версии интерфейса I2C (двухпроводная двунаправленная шина) фирмы *Philips*. Этот интерфейс позволяет объединить вместе до 128 различных устройств с помощью двунаправленной шины, состоящей из линии тактового сигнала (SCL) и линии данных (SDA).
Интерфейс JTAG был разработан группой ведущих специалистов по проблемам тестирования электронных компонентов *(Joint Test Action Group)* и был зарегистрирован в качестве промышленного стандарта *IEEE Std 1149.1-1990*. Четырехпроводной интерфейс JTAG используется для тестирования печатных плат, внутрисхемной отладки, программирования микроконтроллеров. Многие микроконтроллеры семейства *Mega* имеют совместимый с *IEEE Std 1149.1* интерфейс JTAG или *debugWIRE* для встроенной отладки. Кроме того, все микроконтроллеры *Mega* с флэш-памятью емкостью 16 кбайт и более могут

программироваться через интерфейс JTAG. Тактовый генератор вырабатывает импульсы для синхронизации работы всех узлов микроконтроллера. Внутренний тактовый генератор *AVR* может запускаться от нескольких источников опорной частоты (внешний генератор, внешний кварцевый резонатор, внутренняя или внешняя RC-цепочка). Минимальная допустимая частота ничем не ограничена (вплоть до пошагового режима). Максимальная рабочая частота определяется конкретным типом микроконтроллера и указывается *Atmel* в его характеристиках, хотя практически любой *AVR* микроконтроллер с заявленной рабочей частотой, например, в 10 МГц при комнатной температуре легко может быть "разогнан" до 12 МГц и выше. Система реального времени RTC реализована во всех микроконтроллерах *Mega* и в двух кристаллах *«classic»* - *AT90(L)S8535*. Таймер/счетчик RTC имеет отдельный предделитель, который может быть программным способом подключен или к источнику основной тактовой частоты, или к дополнительному асинхронному источнику опорной частоты (кварцевый резонатор или внешний синхросигнал). Для этой цели зарезервированы два вывода микросхемы. Внутренний осциллятор оптимизирован для работы с внешним "часовым" кварцевым резонатором 32,768 кГц.

Микроконтроллер *Intel i8051* выполнен на основе высокоуровневой n-МОП технологии. Через четыре программируемых параллельных порта ввода/вывода и один последовательный порт микроконтроллер взаимодействует с внешними устройствами. Основу структурной схемы МК образует внутренняя двунаправленная восьмибитная шина, которая связывает между собой основные узлы и устройства микроконтроллера: резидентную память программ (RPM), резидентную память данных (RDM), арифметико-логическое устройство (ALU), блок регистров специальных функций, устройство управления (CU) и порты ввода/вывода (P0 − P3). На рисунке 1.15 представлена структурная схема микроконтроллера *Intel 8051*.

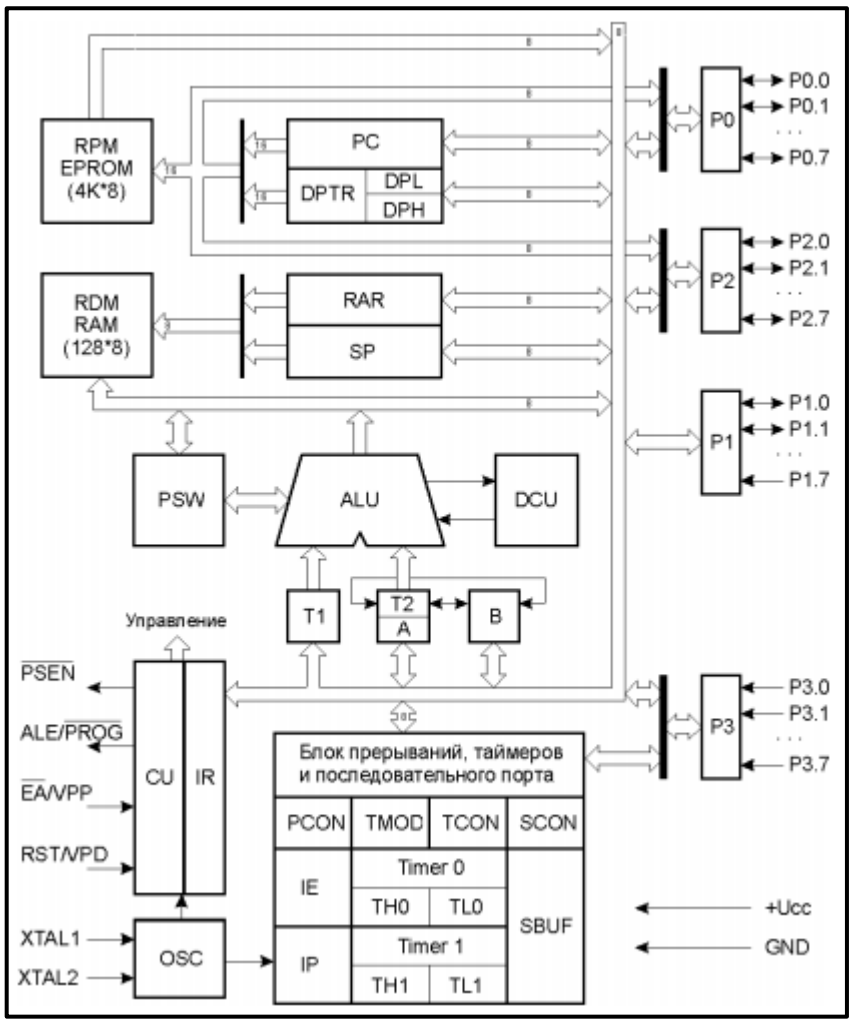


Рисунок 1.15 – Структурная схема микроконтроллера Intel 8051

Восьмиразрядное АЛУ предназначено для выполнения арифметических операций сложения, вычитания, умножения и деления; логических операций «И», «ИЛИ», «исключающее ИЛИ», а также операций циклического сдвига, сброса, инвертирования и т.п. К входам АЛУ подключены программно-недоступные регистры T1 и T2, предназначенные для временного хранения операндов, блок десятичной коррекции (DCU) и блок формирования признаков результата операции (PSW). Простейшая операция сложения используется в ALU для инкрементирования содержимого регистров, продвижения регистра-указателя данных (RAR) и автоматического вычисления следующего адреса резидентной памяти программ. Простейшая операция вычитания используется

в ALU для декрементирования регистров и сравнения переменных. Простейшие операции автоматически образуют “тандемы” для выполнения таких операций, как, например, инкрементирование 16-битных регистровых пар. В ALU реализуется механизм каскадного выполнения простейших операций для реализации сложных команд. Так, например, при выполнении одной из команд условной передачи управления, по результату сравнения в ALU трижды инкрементируется счётчик команд (PC), дважды производится чтение из RDM, выполняется арифметическое сравнение двух переменных, формируется 16-битный адрес перехода и принимается решение о том, делать или не делать переход по программе. Все перечисленные операции выполняются всего лишь за две Мкс. Важной особенностью ALU является его способность оперировать не только байтами, но и битами. Отдельные программно-доступные биты могут быть установлены, сброшены, инвертированы, переданы, проверены и использованы в логических операциях. Эта способность достаточно важна, поскольку для управления объектами часто применяются алгоритмы, содержащие операции над входными и выходными булевыми переменными, реализация которых средствами обычных микропроцессоров сопряжена с определенными трудностями. Таким образом, ALU может оперировать четырьмя типами информационных объектов: булевыми (один бит), цифровыми (четыре бита), байтными (восемь бит) и адресными(16 бит). В ALU выполняется 51 различная операция пересылки или преобразования этих данных. Так как используются 11 режимов адресации (семь для данных и четыре для адресов), то путем комбинирования операции и режима адресации базовое число команд 111 расширяется до 255 из 256 возможных при однобайтном коде операции.

Программатор *PICkit 2* - средство разработки, поддерживающее программирование большинства микроконтроллеров, микросхем памяти и *KeeLOQ* производства компании *Microchip Technology Inc*.

Светодиоды состояния отображают статус программатора/отладчика PICkit 2. Power (зеленый светодиод) показывает, что *PICkit 2* подключен к USB порту.

Target (желтый светодиод) показывает, что *PICkit 2* выдает питание на целевое устройство. Busy (красный светодиод) показывает, что *PICkit 2* занят и выполняет такие функции как программирование, проверку и т.п.

На рисунке 1.16 изображена схема подключения микроконтроллера *PIC16F84A* к программатору *PICKIT2.*

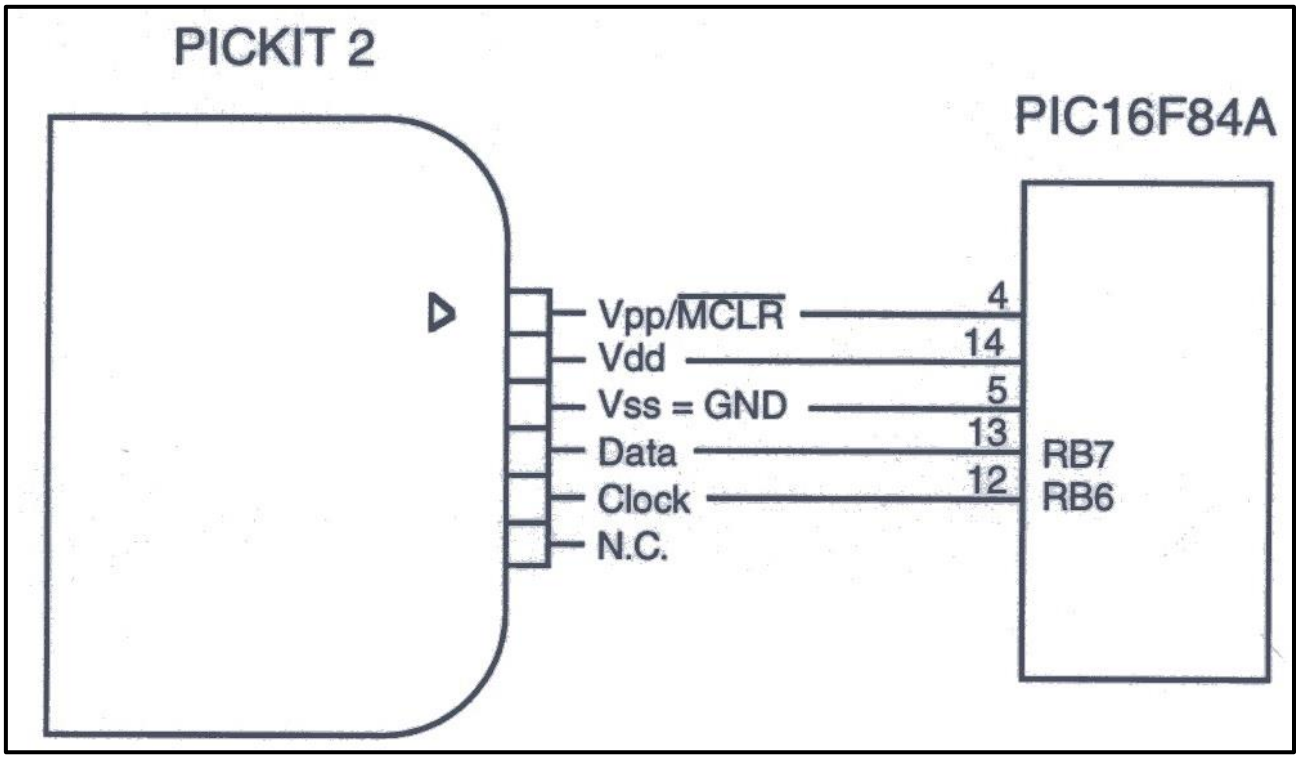


Рисунок 1.16 – Подключение микроконтроллера *PIC16F84A* к программатору *PICKIT2*

PICkit 2 Programmer – это программная оболочка для работы с программатором *PICkit 2* и его клонами, основная функция которого – «прошивка» микроконтроллеров *PIC*. Программа поддерживает режимы чтения, записи, стирания прошивки, а также её проверки после процесса записи. Также с помощью программы можно восстановить калибровочную константу для микроконтроллеров с внутренним RC-генератором, если она была случайно «затёрта». Программа *PICkit 2 Programmer* совместно с программатором *PICkit 2* может использоваться в режиме логического

анализатора или USB-UART преобразователя. Интерфейс программы интуитивно понятен, что важно для тех, кто только недавно начал изучение микроконтроллеров. На рисунке 1.17 изображен интерфейс *Pickit 2 Programmer*.

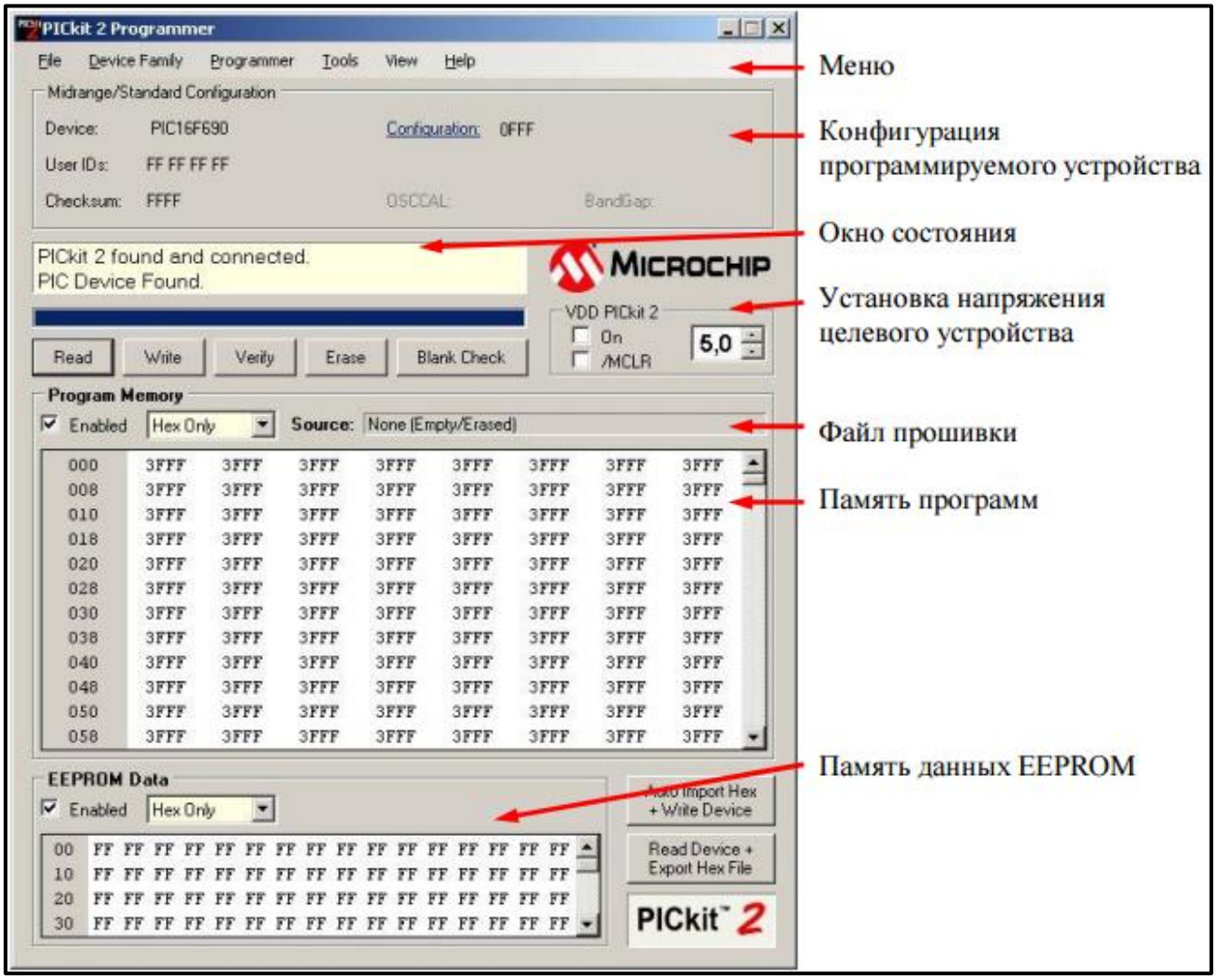


Рисунок 1.17 – Интерфейс PICkit 2 Programmer

Программатор *Fun-Card* предназначен для работы под управлением программы *ICProg*. Программатор подключается к LPT-порту. Разъем устанавливается непосредственно на плату программатора, кроме того, на плате предусмотрена кроватка для программирования контроллера *AT90S2313*, а также выведены сигналы SCK, MOSI/MISO и Reset. Программируемая

микросхема может брать питание с порта LPT, в этом случае, на выводах 2, 3, 4 порта должны быть установлены единицы, а вывод 2 разъёма ISP должет быть подключен к выводу Vcc микросхемы. Некоторые порты могут не потянуть такой нагрузки, в этом случае следует использовать внешний источник питания (пять В). Источником тактовых импульсов для микросхемы также может служить LPT порт. В этом случае вывод 3 разъёма ISP (LED) должен быть подключен к выводу XTAL 1 программируемой микросхемы. Естественно, программа программатора на PC должна понимать эти режимы работы, для работы с этой схемой нужно воспользоваться программой *IC-Prog*, где при выборе типа программатора следует установить *«Fun-Card Programmer»*. На рисунке 1.18 показана схема программатора *Fun-Card.*

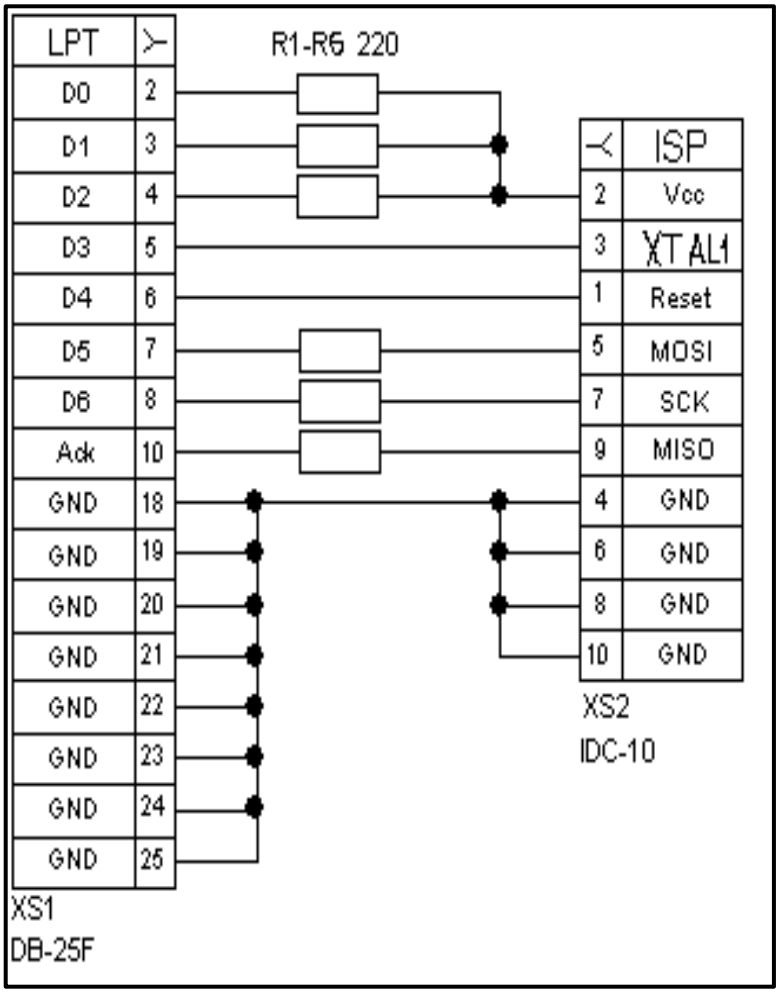


Рисунок 1.18 – Схема программатора *Fun-Card*

AVR910 – весьма известный от *Atmel*, давший название целому классу устройств. Сейчас под *AVR910* понимают как правило протокол, по которому происходит обмен данными между компьютером и программатором. В сети на данный момент можно найти несколько вариантов таких программаторов, различающихся способом реализации интерфейсной части. Традиционно все эти программаторы собираются на основе микроконтроллера *AT90S2313*. На схеме представлен программатор, способный работать как через COM, так и через USB. Переключение типа интерфейса происходит при помощи джампера J1. При работе через USB питание программатора осуществляется непосредственно от этого порта компьютера, причем в этом режиме имеется полная гальваническая развязка программатора от компьютера, более того, при замыкании перемычки J2 программируемое устройство может питаться от программатора (до 100 мА). При работе через COM-порт развязка отсутствует, а питание программатора осуществляется, как обычно, от программируемого устройства. Интерфейс USB реализован на микросхеме *FT232BM* в стандартной схеме включения, в качестве согласователя уровней для COM-порта применена *MAX232*. Схема обеспечивает уровень выходного напряжения, используемый в *RS-232* (приблизительно ± 7.5 В), преобразуя входное напряжение плюс пять Вольт при помощи внутреннего зарядового насоса на внешних конденсаторах. Это упрощает реализацию *RS-232* в устройствах, работающих на напряжениях от нуля до пяти В, так как не требуется усложнять источник питания только для того, чтобы использовать *RS-232*. Функциональность и цоколевка микросхемы стала стандартом де-факто и её аналоги (с другой маркировкой) выпускаются множеством производителей полупроводников. На рисунке 1.19 изображена схема программатора *AVR910* с универсальным COM/USB интерфейсом.

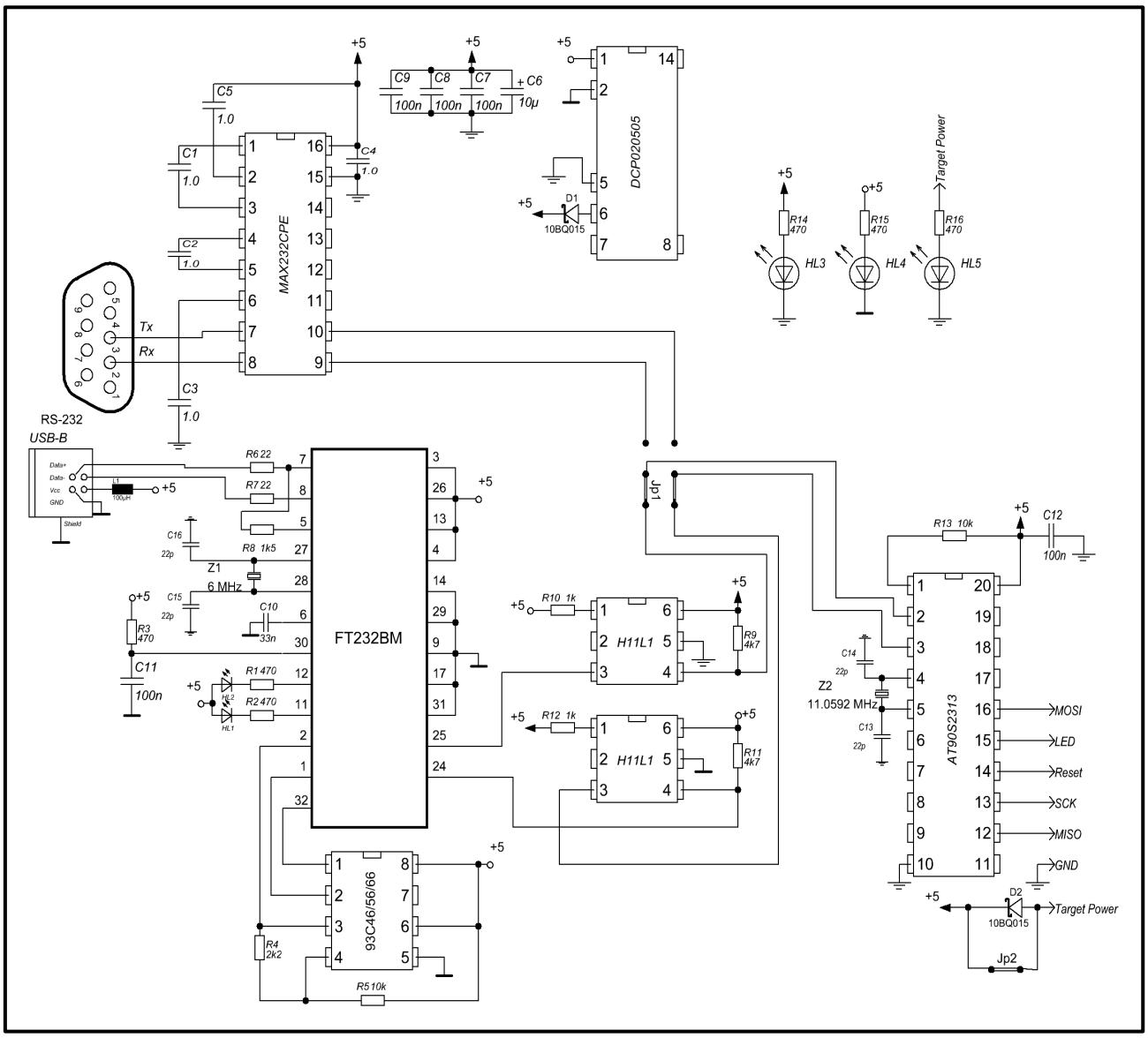


Рисунок 1.19 - Схема программатора *AVR910* с универсальным COM/USB интерфейсом

Программный пакет *AVR Studio* разрабатывается с 2004 года. Начиная с версии 6.0, программа сменила название на *Atmel Studio*. Программа позволяет работать как на ассемблере, так и на *C/C++*. Содержит в себе мастер проектов, виртуальный симулятор, редактор исходного кода, модуль внутрисхемной отладки и интерфейс командной строки. Поддерживает компилятор *GCC* и плагин *AVR RTOS* (операционной системы реального времени). Пользователи могут выбрать наиболее оптимальные для их проекта способы кодирования.

Визуальные инструменты позволяют ускорить написание программы. Благодаря связке программных пакетов *Atmel Studio* и *Proteus* от фирмы *Labcenter Electronics* возможно программирование микроконтроллеров без наличия какой-либо материальной базы. *Atmel Studio* по праву считается лучшей средой создания приложений для контроллеров *AVR*. Последняя версия *Atmel Studio* поддерживает все существующие на сегодняшний момент восьмибитные, 32-битные *AVR, SAM3* и *SAM4* микроконтроллеры и включает в себя более 1100 проектов с примерами. Также доступны старые версии программы. Интерфейс полностью англоязычный и официального русификатора нет. Программа не понимает русских символов, поэтому названия работ должны быть написаны с английской транскрипцией. *Atmel Studio* работоспособна в операционных системах *Windows 9x / ME / NT / 2000 / XP / VISTA / 7*. На рисунке 1.20 изображен интерфейс *Amtel Studio.*

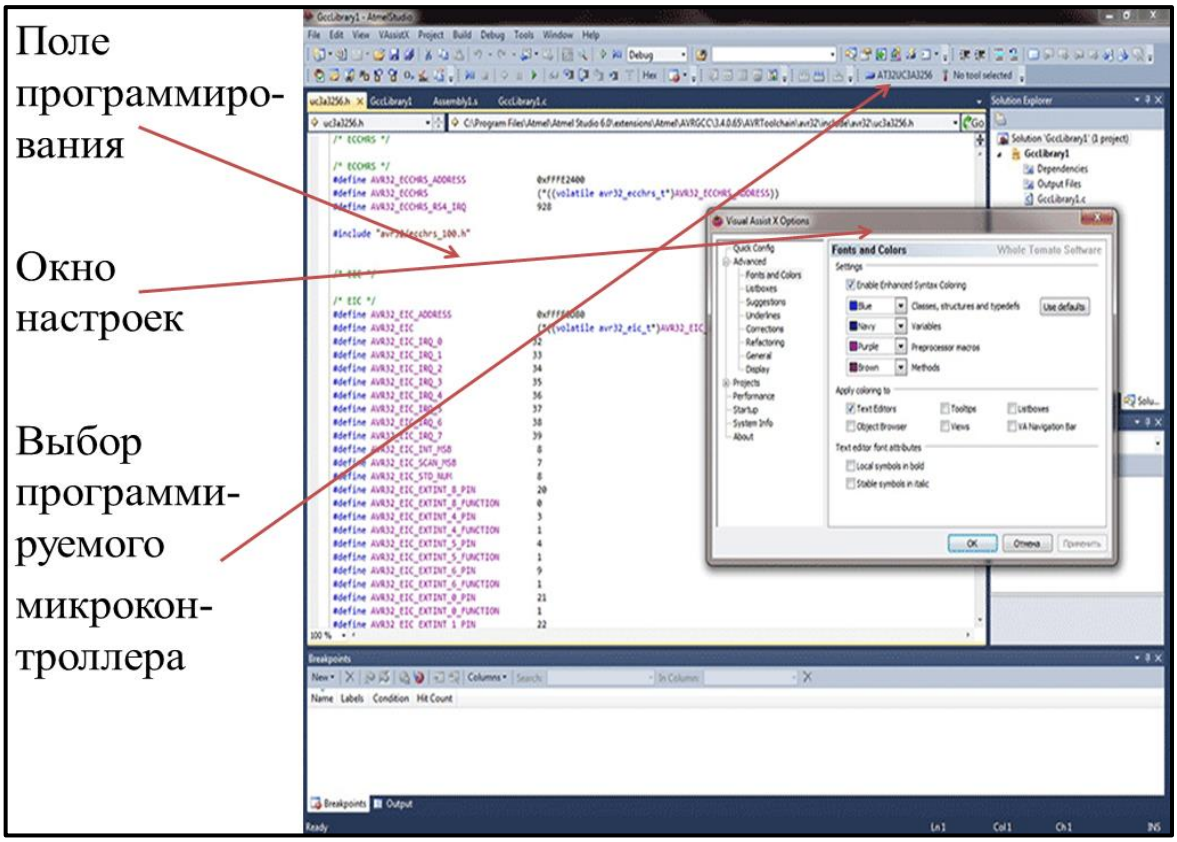


Рисунок 1.20 – Интерфейс *Amtel Studio*

Программная оболочка *MPLAB* обладает широкими возможностями для написания исходного кода программы, дальнейшей отладки текста с исправлением ошибок и предупреждений и финальной оптимизации проекта. Для того чтобы реализовать столь обширные функций в *MPLAB* входят следующие модули:

- менеджер проектов *(Project Manager)* для управления различными файлами рабочих групп;

- редактор *(Editor)*, в котором авторы создают свои программы и поправляют их;

- встроенный отладчик микроконтроллеров *PIC16F87X MPLAB ICD*;

- симулятор *MPLAB-SIM*, пошагово моделирующий работу программы в микросхеме;

- эмуляторы *MPLAB-ICE, PICMASTER-CE* и *PICMASTER* для виртуального представления поведения контроллера на аппаратуре разработчика в режиме реального времени;

- целый ряд компиляторов *(MPLAB C-17, MPLAB C-18, MPASM, MPLINK)*, преобразующих исходный код, написанный на разных языках программирования (ассемблер, Си);

- редактор библиотек *MPLIB*;

- программаторы *PRO MATE* и *PICSTART plus*, обеспечивающие перенос программ во внутреннюю память микроконтроллеров.

Также возможно подключение дополнительных модулей, разработанных сторонними специалистами. Благодаря встроенной системе помощи программа довольно проста в изучении, разумеется, при наличии определенных знаний. А производитель PIC-контроллеров *Microchip*, разработавший *MPLAB,* осуществляет прекрасную поддержку своих продуктов. В папке, где размещается *MPLAB*, по пути *\template\code* лежат файлы-шаблоны для проектов, разрабатываемых на ассемблере, с которых удобно начинать работу. Необходимо помнить, что для каждого проекта *MPLAB* создает целый ряд вспомогательных файлов, поэтому если в одном месте будут находиться

несколько работ, то можно перепутать их. На рисунке 1.21 изображен интерфейс *MPLAB.*

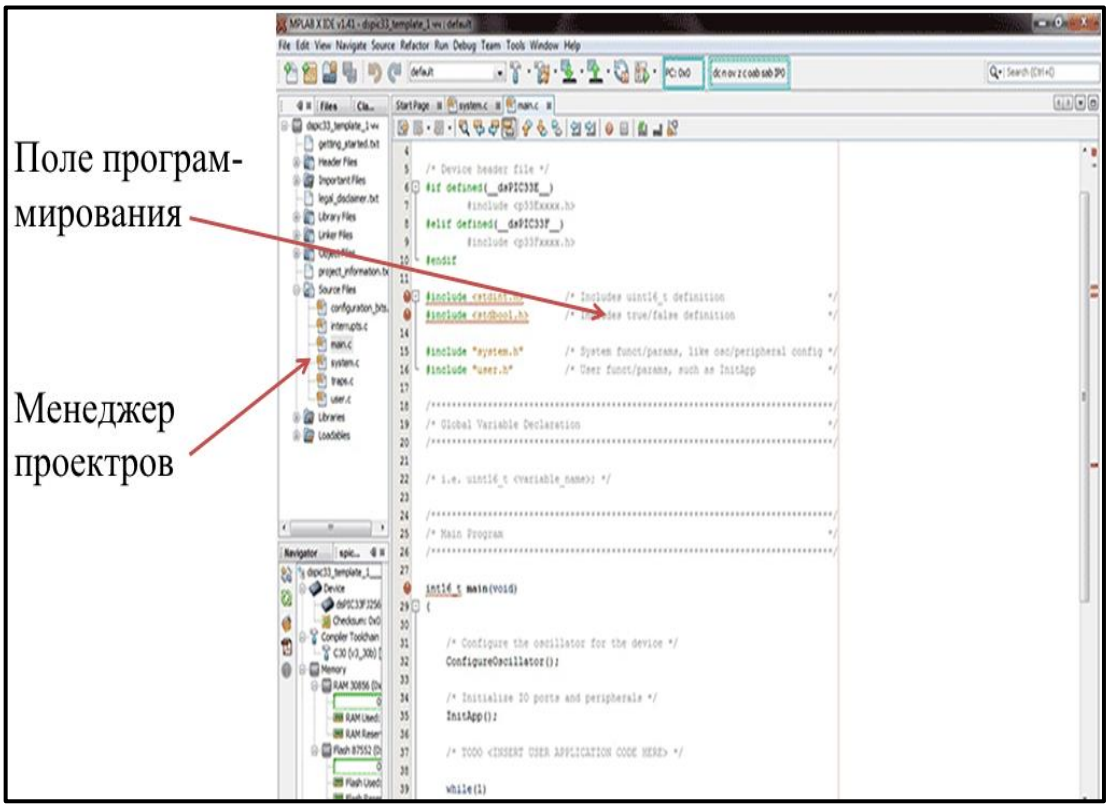


Рисунок 1.21 – Интерфейс *MPLAB*

Приложение *PICPgm Development Programmer* имеет простой, дружелюбный графический интерфейс и предназначено для работы с микроконтроллерами семейства *PIC*. Основное окно пакета состоит из меню, панели инструментов, рабочей части со вкладками и строки состояния. Программное обеспечение работает с flash-памятью, внутренним *EEPROM* и конфигурационными битами согласно данным из hex-файла. В *PICPgm* имеются стандартные для подобного рода программ функции: чтение содержимого микроконтроллера и запись его в hex-файл, очистка памяти чипа, верификация, автоопределение модели программатора и микроконтроллера, проверка объема свободной памяти контроллера. Поддерживаются методики низковольтного и внутрисхемного (ISP) программирования. Кроме того могут

быть сконфигурированы выводы программатора. На рисунке 1.22 представлен интерфейс *PICPgm.*

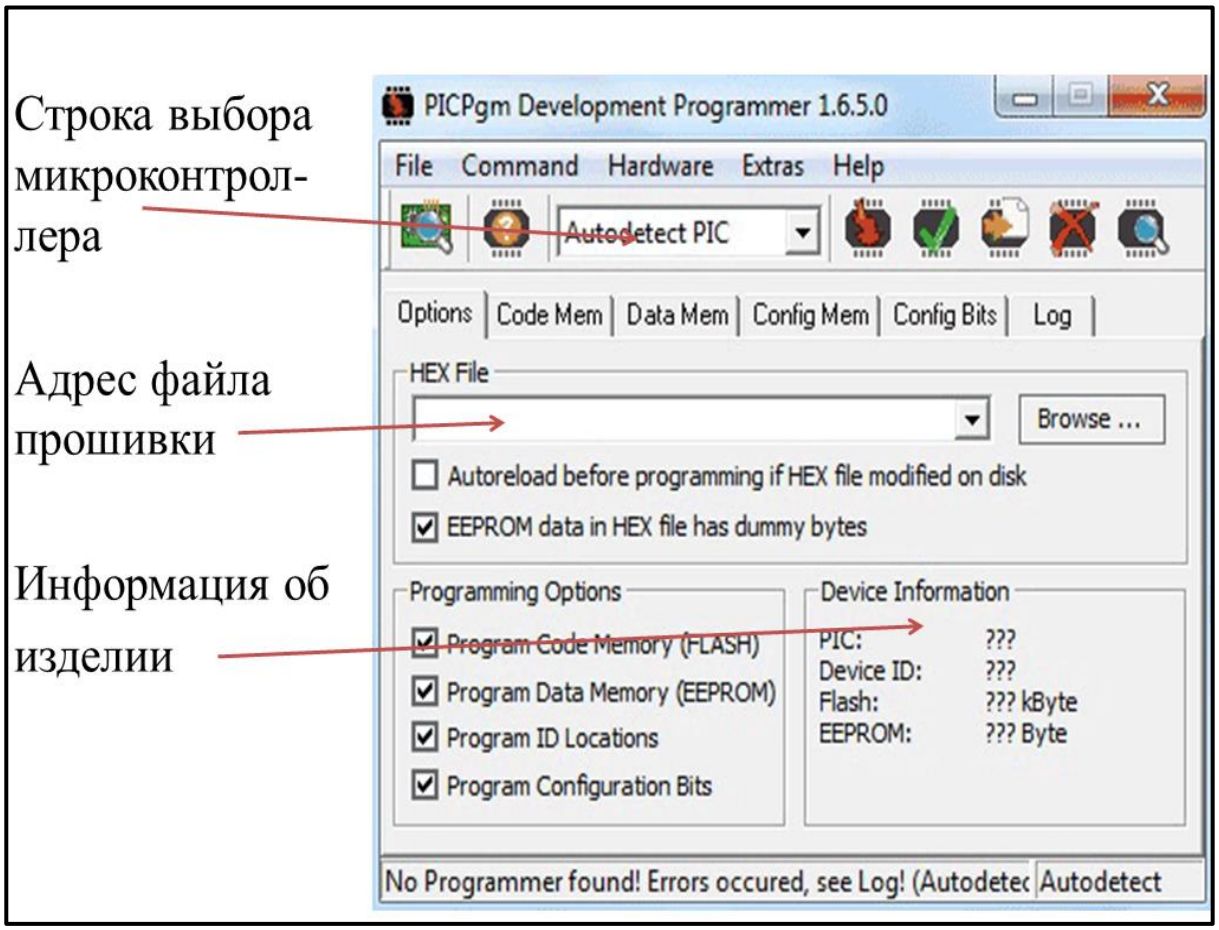


Рисунок 1.22 – Интерфейс *PICPgm*

1.4 Вывод по главе 1

Рассмотрены наиболее популярные отечественные автоматизированные системы для автотранспорта, а именно БК АМК211501, БК таких автомобилей, как *Lada Kalina, Vesta* и способы их управления. Рассмотрены наиболее популярные семейства микроконтроллеров, таких как *PIC, ARM Cortex, AVR* и *Intel 8051.*

Семейство *ARM Cortex* является завершенным процессорным ядром, которое объединяет стандартное ЦПУ и системную архитектуру. Микроконтроллеры *STM32* выполнены на основе профиля *Cortex-M3*, которое специально разработано для применений, где необходимы развитые системные ресурсы и, при этом, малое энергоптребление. Они характеризуются настолько низкой стоимостью, что могут конкурировать с традиционными восьми и 16-битными микроконтроллерами. *Cortex-M3* является стандартизованным микроконтроллерным ядром, которое помимо ЦПУ, содержит все остальные составляющие основу микроконтроллера элементы (система прерываний, системный таймер *SysTick*, отладочная система и карта памяти). Четырех гигабайтное адресное пространство *Cortex-M3* разделено на четко распределенные области кода программы, статического ОЗУ, устройств ввода-вывода и системных ресурсов. *Cortex-M3* выполнено по Гарвардской архитектуре и имеет несколько шин, позволяющие выполнять операции параллельно.

Семейство *PIC* микроконтроллеров имеют гарвардскую архитектуру и производятся американской компанией *Microchip Technology Inc*. *PIC16F84* относится к семейству КМОП микроконтроллеров. Отличается тем, что имеет внутреннее 1024 x 14 бит *EEPROM* для программ, восьмибитовые данные и 64байт EEPROM памяти данных. При этом отличаются низкой стоимостью и высокой производительностью. Высокая нагрузочная способность (25 мА максимальный входной ток, 20 мА максимальный выходной ток) линий ввода/вывода упрощают внешние драйверы и, тем самым, уменьшается общая

стоимость системы. Разработки на базе контроллеров *PIC16F84* поддерживается ассемблером, программным симулятором, внутрисхемным эмулятором (только фирмы *Microchip*) и программатором. Встроенный автомат программирования *EEPROM* кристалла *PIC16F84* позволяет легко подстраивать программу и данные под конкретные требования даже после завершения ассемблирования и тестирования. Эта возможность может быть использована как для тиражирования, так и для занесения калибровочных данных уже после окончательного тестирования.

Микроконтроллеры *AVR* примерно в четыре раза быстрее, чем *PIC*. Кроме того, они потребляют немного энергии и могут работать в четырех режимах экономии энергии. *AVR* принадлежат к типу RISC-микроконтроллеров. Архитектура RISC *(Complex Instruction Set Computers)* означает, что набор инструкций, которые может выполнять процессор устройства, является ограниченным, но, в то же время, подобная архитектура дает преимущество в скорости. В микроконтроллерах *AVR* реализована Гарвардская архитектура, в соответствии с которой разделены не только адресные пространства памяти программ и памяти данных, но и шины доступа к ним. Каждая из областей памяти данных (оперативная память и *EEPROM*) также расположена в своем адресном пространстве.

Микроконтроллер *i8051* выполнен на основе высокоуровневой n-МОП технологии. Через четыре программируемых параллельных порта ввода/вывода и один последовательный порт микроконтроллер взаимодействует с внешними устройствами. Основу структурной схемы МК образует внутренняя двунаправленная восьмибитная шина, которая связывает между собой основные узлы и устройства микроконтроллера: резидентную память программ, резидентную память данных, арифметико-логическое устройство, блок регистров специальных функций, устройство управления и порты ввода/вывода. Может оперировать четырьмя типами информационных объектов: булевыми (один бит), цифровыми (четыре бита), байтными (восемь бит) и адресными(16 бит).

Программатор *PICkit 2* - средство разработки, поддерживающее программирование большинства микроконтроллеров, микросхем памяти и *KeeLOQ* производства компании *Microchip Technology Inc. PICkit 2 Programmer* – это программная оболочка для работы с программатором *PICkit 2* и его клонами, основная функция которого – «прошивка» микроконтроллеров *PIC*. Программатор *Fun-Card* предназначен для работы под управлением программы *ICProg. AVR910* – весьма известный программатор от *Atmel*, давший название целому классу устройств. Сейчас под *AVR910* понимают как правило протокол, по которому происходит обмен данными между компьютером и программатором. *Atmel Studio* содержит в себе мастер проектов, виртуальный симулятор, редактор исходного кода, модуль внутрисхемной отладки и интерфейс командной строки. *Atmel Studio* считается лучшей средой создания приложений для контроллеров *AVR*. Производитель PIC-контроллеров *Microchip*, разработавший *MPLAB*, осуществляет прекрасную поддержку своих продуктов. В папке, где размещается *MPLAB*, по пути *\template\code* лежат файлы-шаблоны для проектов, разрабатываемых на ассемблере, с которых удобно начинать работу. Приложение *PICPgm Development Programmer* имеет простой, дружелюбный графический интерфейс и предназначено для работы с микроконтроллерами семейства PIC.

Рассмотрены наиболее известные среды программирования микроконтроллеров и программаторов.

В рамках главы 1 достигнуты следующие задачи:

1) обзор и анализ существующих автоматизированных систем отечественного автотранспорта;
2) обзор и анализ существующих семейств микроконтроллеров и сред программирования для каждого из них.

Глава 2. Проектирование информационной системы «датчик дождя»

2.1 Коммуникационные интерфейсы платы *Arduino Uno*

Модуль *Arduino UNO* имеет средства для связи с компьютером, с другой платой *UNO* или с другими микроконтроллерами. На плате существует интерфейс UART с логическими уровнями TTL (пять *B*), связанный с выводами 0 (RX) и 1(TX). Микросхема *ATmega16U2* на плате связывает UART интерфейс с USB портом компьютера. При подключении к порту компьютера, появляется виртуальный COM порт, через который программы компьютера работают с *Arduino*. Прошивка *ATmega16U2* использует стандартные драйверы USB-COM и установка дополнительных драйверов не требуется. В интегрированную среду программного обеспечения *Arduino IDE* включен монитор обмена по последовательному интерфейсу, который позволяет посылать и получать с платы простые текстовые данные. На плате есть светодиоды RX и TX, которые индицируют состояние соответствующих сигналов для связи через USB (но не для последовательного интерфейса на выводах 0 и 1). Микроконтроллер *ATmega328* также поддерживает коммуникационные интерфейсы I2C (TWI) и SPI. Uart - простой последовательный интерфейс для организации информационного канала обмена микроконтроллера с внешним миром. Способен работать в дуплексном режиме (одновременная передача и прием данных). Поддерживает протокол стандарта *RS-232*, что обеспечивает возможность организации связи с персональным компьютером. При передаче микросхема UART преобразует параллельный код в последовательный и передает его побитно в линию, обрамляя исходную последовательность битами старта, останова и контроля. При приеме данных UART преобразует последовательный код в параллельный. Непременным условием правильной передачи (приема) является одинаковая скорость работы приемного и передающего UART, что обеспечивается стабильной частотой кварцевого резонатора. Основным преимуществом последовательной передачи является

возможность пересылки данных на большие расстояния, как правило, не менее 30 метров. С помощью UART можно связать микроконтроллер и компьютер. Для этого следует использовать COM-порт компьютера. У UART интерфейса логические уровни нуль и пять В, а в компьютере логические уровни в интерфейсе *RS-232* могут быть от минус 25 до минус трех В и от плюс трех до плюс 25 В. Для этого применяют специальный преобразователь уровней в нашем случае это микросхема *ATmega16U2*. Протокол предполагает двусторонний обмен данными между двумя устройствами. При этом оба устройства являются равноправными, нет деления на главного и подчиненного. Для организации обмена данными используется всего две линии. По одной линии данные передаются от первого устройства ко второму, а по другой - в обратном направлении. У МК задействуются два пина. Обозначаются они соответственно Tx - передающий, и Rx - принимающий. Линии передачи соединяют Tx устройства 1 с Rx устройства 2 и Rx устройства 1 с Tx устройства 2. На рисунке 2.1 показана схема подключения устройств через интерфейс Uart.

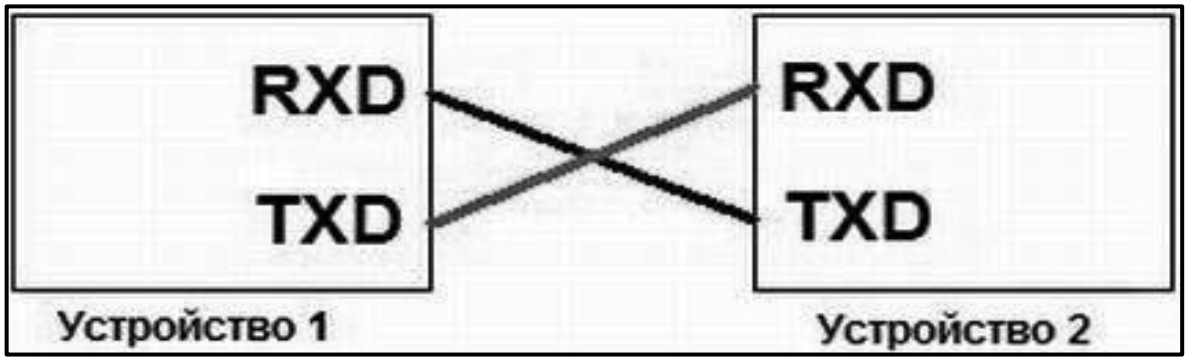


Рисунок 2.1 – Схема подключения устройств через интерфейс Uart

В нашей стране в 1982г. введен ГОСТ 18145-81 стандарт соединения оборудования.

Подключение платы *Arduino Uno* к ПК выполняется через USB-кабель. Соединив консоль *Arduino* с ПК на ней загорится светодиод «ON», и начнет

мигать второй «L». Это означает, что через кабель подано питание и микроконтроллер начал выполнять предустановленную на заводе программу Blink. Далее следует узнать какой номер COM-порта присвоил компьютер плате *Arduino*. Номер COM-порта можно узнать в диспетчере устройств, в вкладке «Порты(COM и LPT)». На рисунке 2.2 изображен диспетчер устройств вкладка «Порты».

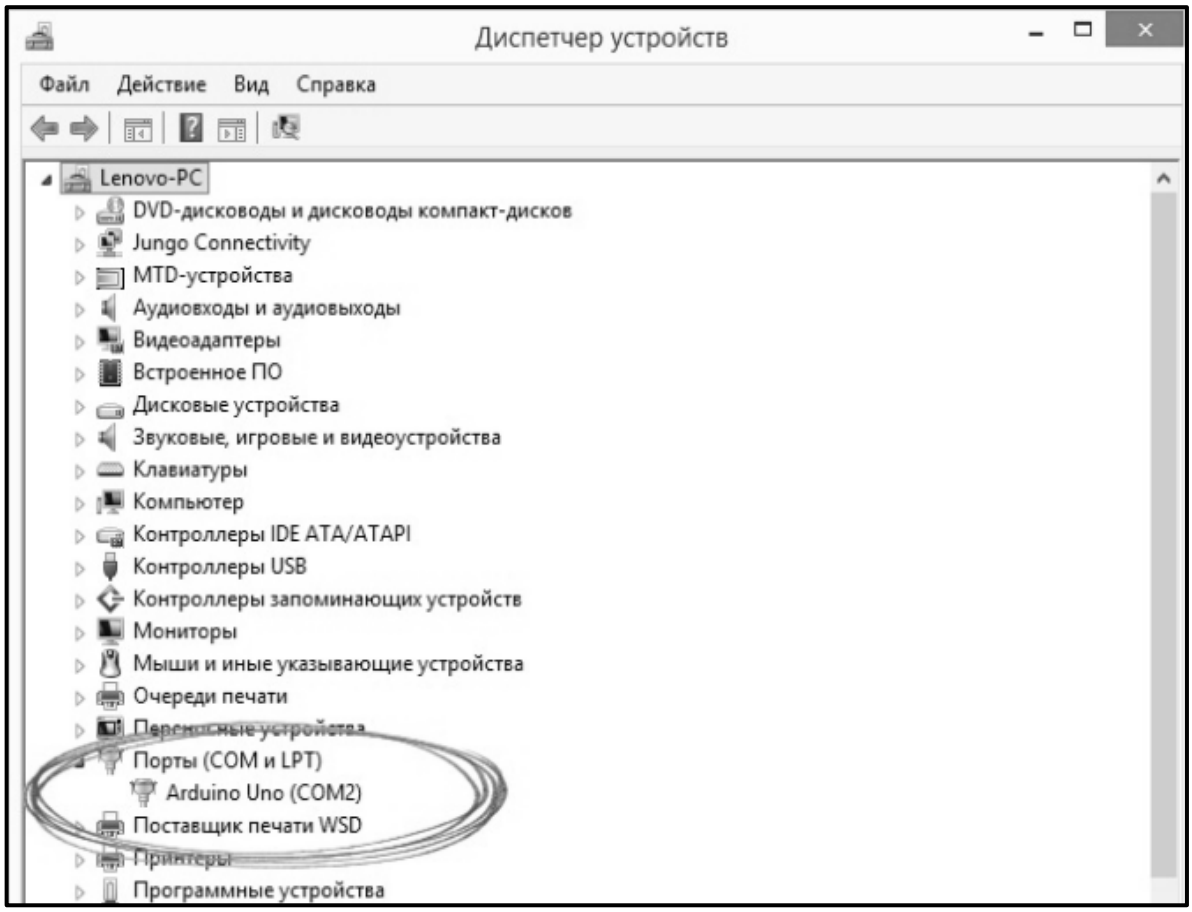


Рисунок 2.2 – Диспетчер устройств, вкладка «Порты»

Если высветилось наше устройство Arduino, значит ОС распознала плату, нашла для нее подходящий USB драйвер и присвоила номер его интерфейсу. Напряжение интерфейса USB всех версий 4,5-5,5 *В*, это значит что можно использовать любую из них, более того на плате имеется предохранитель, который размыкает соединение при силе тока более 500 мА.

2.2 Проектирование автоматизированной информационной системы «датчик дождя»

На рисунке 2.3 представлена функциональная схема современного датчика дождя на четырех оптопарах.

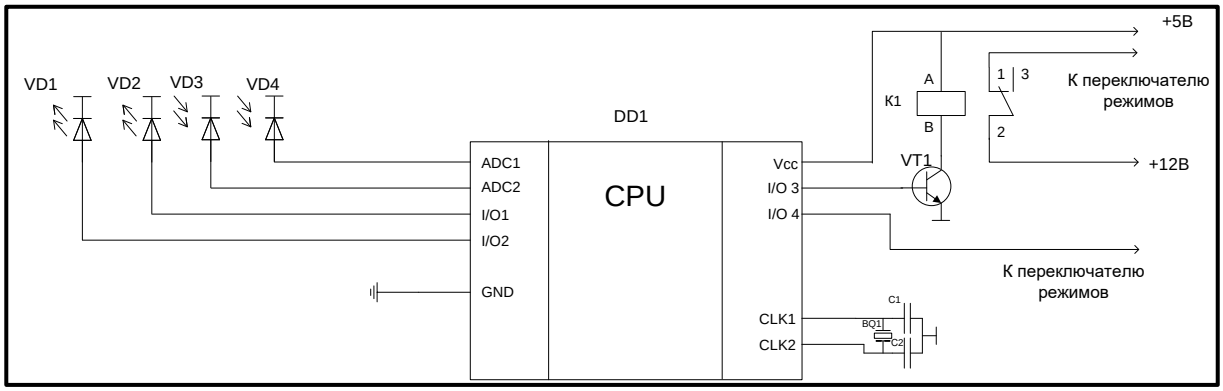


Рисунок 2.3 – функциональная схема датчика дождя

Блок датчиков влажности осуществляет слежение за изменением влажности окружающей среды в процентном соотношении. Блок управления – осуществляет обработку сигналов поступающих от датчиков и формирует управляющие сигналы для управления исполнительным устройством. Блок управления реализуется на базе микропроцессора. Блок согласования осуществляет согласование сигналов от управляющего устройства с бортовой шиной автомобиля. Исполняющее устройство выполняет команды управляющего устройства, в нашем случае роль исполняющего устройства отводится штатным устройствам автомобиля стеклоочистителю и омывателю. Сигналы датчиков поступают на АЦП микроконтроллера, который в соответствии с заложенным алгоритмом производит управление стеклоочистителями автомобиля. На рисунке 2.4 представлена структурная схема датчика дождя.

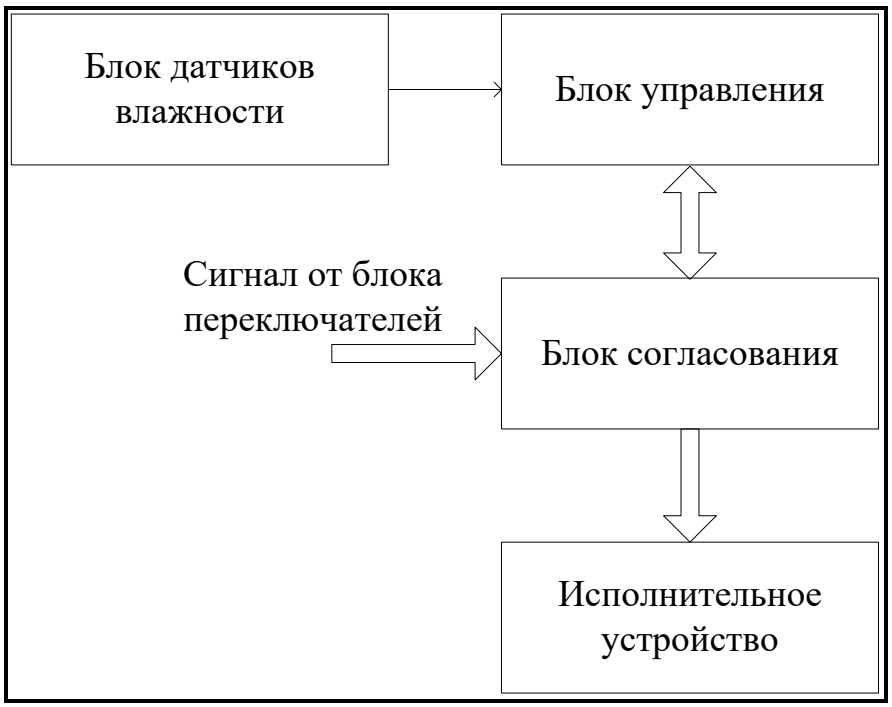
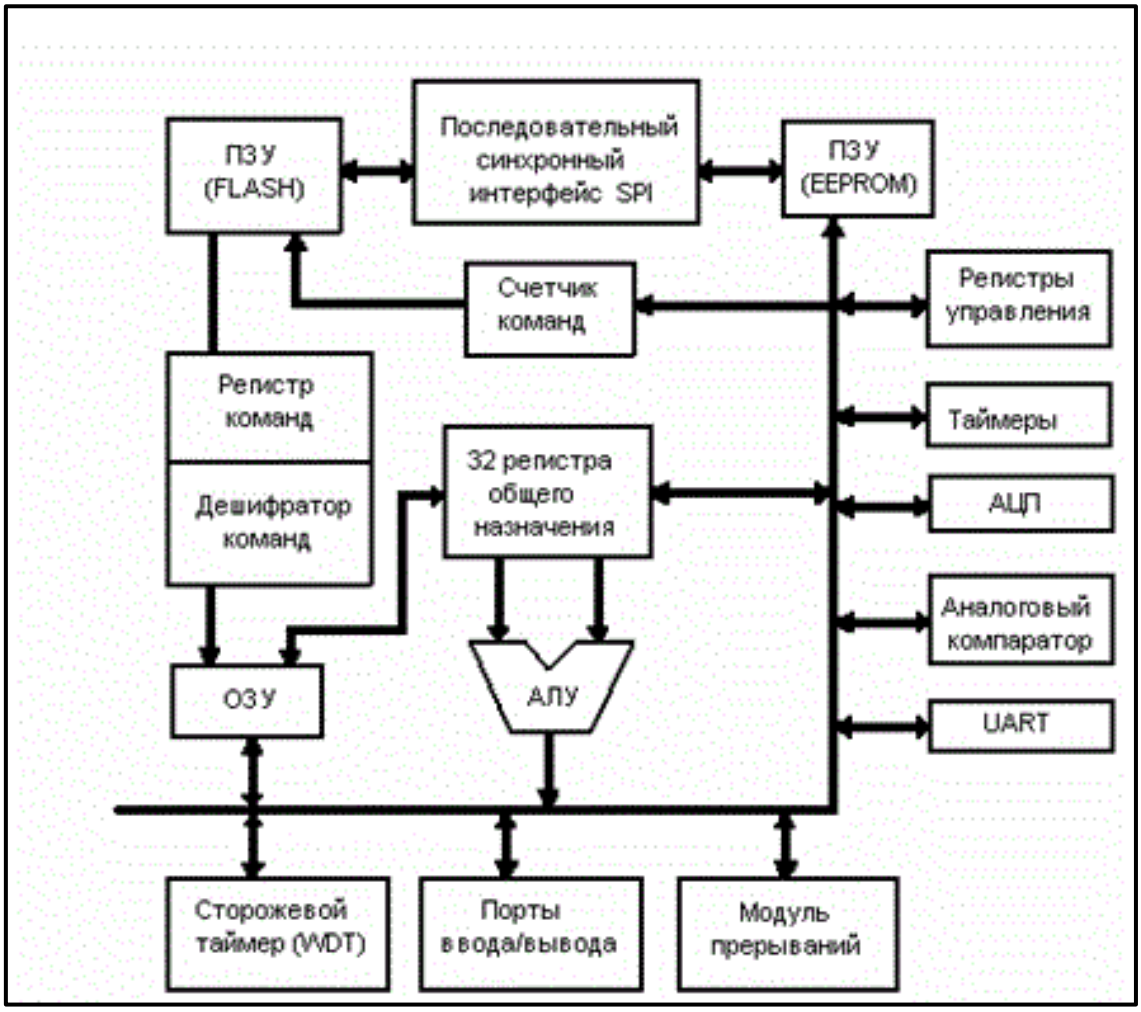


Рисунок 2.4 – Структурная схема датчика дождя

Плата *Arduino Uno* построена на базе *AVR* микроконтроллера. *AVR* функционируют при напряжениях питания от 1,8 до шести В. Ток потребления в активном режиме зависит от величины напряжения питания и частоты, на которой работает микроконтроллер, и составляет менее одного мА для 500 кГц, от пяти до шести мА для пяти МГц и от восьми до девяти мА для частоты 12 МГц. *AVR* могут быть переведены программным путем в один из трех режимов пониженного энергопотребления. Режим холостого хода (IDLE). Прекращает работу только процессор и фиксируется содержимое памяти данных, а внутренний генератор синхросигналов, таймеры, система прерываний и сторожевой таймер продолжают функционировать. Ток потребления не превышает 2,5 мА на частоте 12 МГц. Стоповый режим (POWER DOWN). Сохраняется содержимое регистрового файла, но останавливается внутренний генератор синхросигналов, и, следовательно, останавливаются все функции, пока не поступит сигнал внешнего прерывания или аппаратного сброса. При включенном сторожевом таймере ток потребления в этом режиме составляет

около 80 мкА, а при выключенном - менее одного мкА. (Все приведенные значения справедливы для напряжения питания пять В). Экономичный режим (POWER SAVE). Продолжает работать только генератор таймера, что обеспечивает сохранность временной базы. Все остальные функции отключены. Схема BOD *(Brown-Out Detection)* отслеживает напряжение источника питания. Если схема включена, то при снижении питания ниже некоторого значения она переводит микроконтроллер в состояние сброса. Когда напряжение питания вновь увеличится до порогового значения, запускается таймер задержки сброса. После формирования задержки внутренний сигнал сброса снимается и происходит запуск микроконтроллера. На рисунке 2.5 изображена типовая архитектура микроконтроллеров *AVR.*



На рисунке 2.5 изображена типовая архитектура микроконтроллеров *AVR.*

На рисунке 2.6 показано распределение пинов мк.

```
 9   ;       GP0 (7) - вход управления периодом включения реле дворников.
10   ;       GP1 (6) - выход включения реле щеток.
11   ;       GP2 (5) - выход включения реле омывателя.
12   ;       GP3 (4) - вход включения режима автомата.
13   ;       GP4 (3) - вход включения прерывистого режима.
14   ;       GP5 (2) - выход включения светодиода.
```

Рисунок 2.6 распределение пинов мк

На рисунке 2.7 показан код совместной работы щеток и омывателя.

```
74        bsf     GPIO, 5          ; Включаем светодиод.
75        bsf     GPIO, 2          ; Включаем реле омывателя.
76        call    Delay_1.0
77        call    Delay_0.6
78        bsf     GPIO, 1          ; Включаем реле дворников.
79        call    Delay_1.0
80        call    Delay_1.0
81        call    Delay_0.6
82        bcf     GPIO, 2          ; Выключаем реле омывателя.
83        call    Delay_1.0
84        call    Delay_1.0
85        call    Delay_1.0
86    ;   call    Delay_1.0
87        bcf     GPIO, 1          ; Выключаем реле дворников.
88        bcf     GPIO, 5          ; Выключаем светодиод.
89        return
```

Рисунок 2.7 – код совместной работы щеток и омывателя

Наиболее подходящие среды программирования микроконтроллеров *AVR* на базе платы *Arduino UNO* для начинающего пользователя являются *Arduino IDE,* благодаря возможности питать, программировать и обмениваться

сообщениями с *Arduino UNO* при помощи одного USB кабеля, а также имеет большой набор стандартных библиотек, которые помогут сделать простой проект начинающему пользователю.

Arduino Uno R3 контроллер построен на *ATmega328*. Платформа имеет 14 цифровых вход/выходов, шесть из которых могут использоваться как выходы широтно-импульсной модуляции, шесть аналоговых входов, кварцевый генератор 16 МГц, разъем USB, силовой разъем, разъем ICSP и кнопку перезагрузки. Для работы необходимо подключить платформу к компьютеру посредством кабеля USB, либо подать питание при помощи адаптера AC/DC или батареи. На рисунке 2.8 показан внешний вид платы *Arduino Uno*.

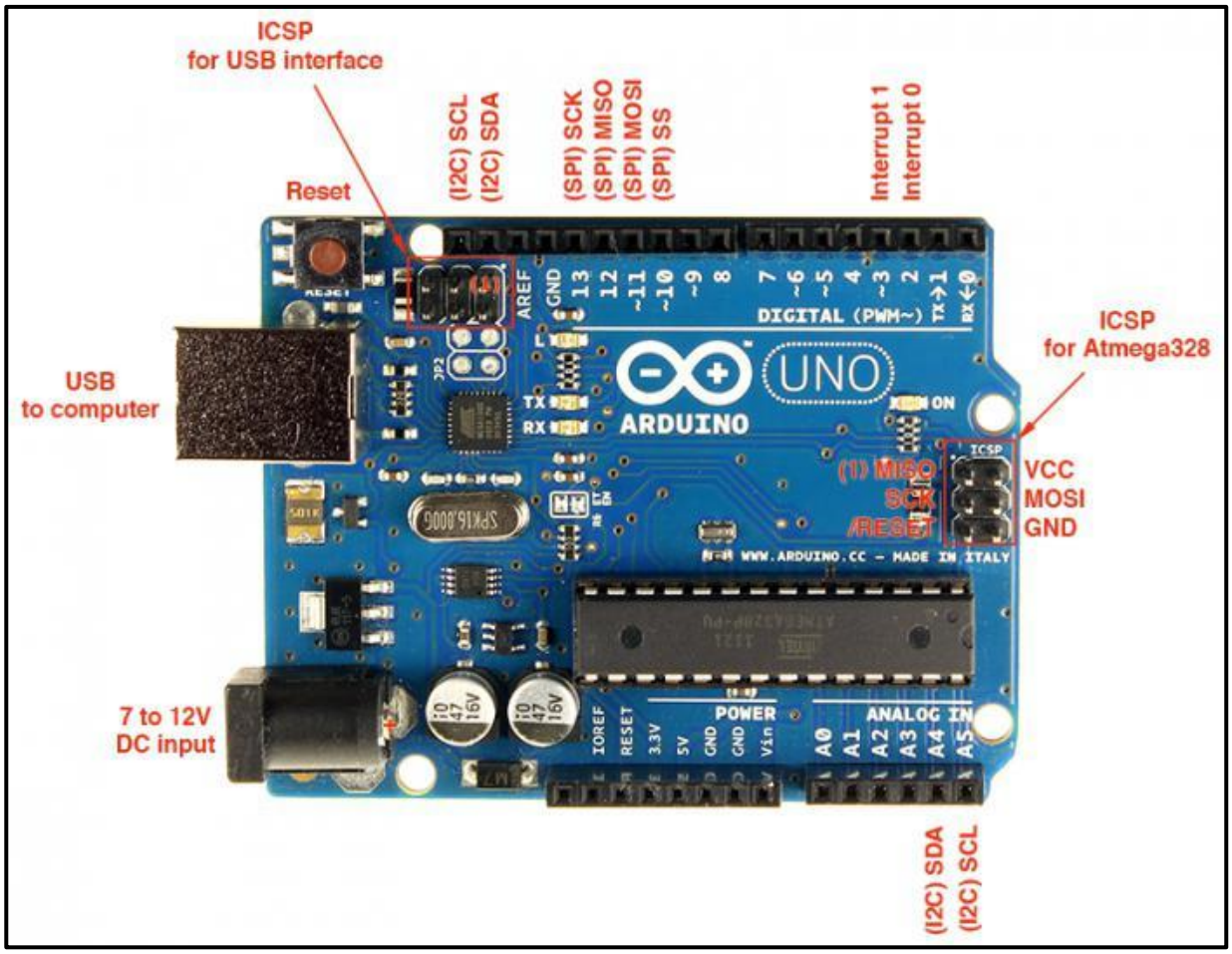


Рисунок 2.8 – Внешний вид платы *Arduino Uno*

Для корректного функционирования *Arduino UNO* драйвер для него должен быть установлен на компьютере, что работает с платой. Выбор драйвера зависит от операционной системы. Существует отдельное ПО для *Arduino UNO*: драйвер *Windows 7, Windows Vista и XP.* С помощью любой аппаратуры, на которую установлены эти операционные системы, можно работать с печатной платой. *Arduino UNO* совместим со всеми компьютерами, выпущенными с 2000г. Диапазон напряжений питания *ATmega328* от двух В до 5,5 В. Напражение бортовой сети автомобиля около 14 В. Для получения стабильного напряжения питания микроконтроллера будет использоваться пятивольтовый стабилизатор напряжения *L4949*. На рисунке 2.9 изображен внешний вид и структурная схема регулятора напряжения *L4949*.

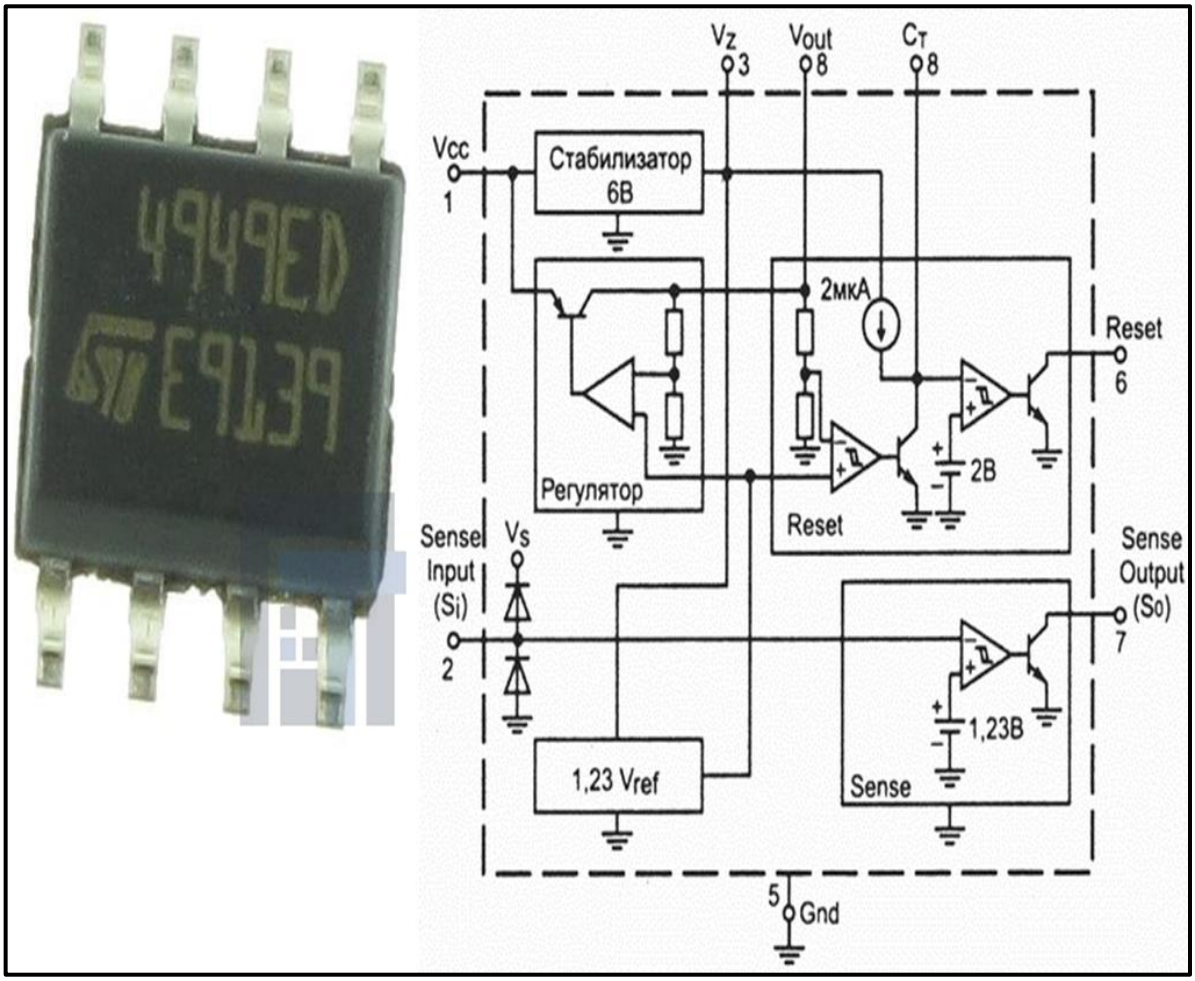


Рисунок 2.9 – Внешний вид и структурная схема регулятора напряжения *L4949*

На выводе шесть микросхемы *L4949* также формируется сигнал начального сброса. При снижении питающего напряжения ниже 10В на входе этой микросхемы, она формирует сигнал аварии, который поступает на вывод микропроцессора.

Характеристики *L4949*:

- минимальное входное напряжение 5 В;
- максимальное входное напряжение 28 В;
- выходное напряжение 5 В;
- рабочая температура от минус 40 до 150 °С.

Питание стабилизатора напряжения берется от бортовой сети автомобиля, а именно с предохранителя на три или пять ампер в блоке предохранителей. В качестве датчика выбран датчик температуры и влажности *DHT11*. Он выполнен из двух частей — емкостного датчика влажности и термистора. Чип, находящийся внутри, выполняет аналого-цифровое преобразование и выдает цифровой сигнал, который можно считать с помощью любого микроконтроллера.

Технические характеристики *DHT11*:

- питание 3-5 В;
- максимально потребляемый ток – 2,5 мА при преобразовании (при запросе данных);
- рассчитан на измерение уровня влажности в диапазоне от 20% до 80%. При этом точность измерений находится в диапазоне 5%;
- измеряет температуру в диапазоне от 0 до 50 градусов с точностью плюс-минус 2%;
- частота измерений не более 1 Гц.

Подключаем плату *Arduino Uno* по USB-кабелю к компьютеру. Скачиваем библиотеку *«DHT.h».* Библиотеку загружаем ссылке

https://github.com/amperka/dht. После загрузки извлекаем содержимое папок *«lib/dht»* в директорию, где хранятся все библиотеки *«C:\Program Files\Arduino\libraries».* После этого запустим *Arduino IDE* и набераем в скетче код. На рисунке 2.10 показана программа считывания данных с *DHT11*.

Рисунок 2.10 - Программа считывания данных с DHT11

Загружаем код в *Arduino UNO* и откроем «Монитор порта». В мониторе видно значение влажности в процентном соотношении и показатели температуры. На рисунке 2.11 показаны значения влажности в процентном соотношении и показатели температуры в «мониторе порта».

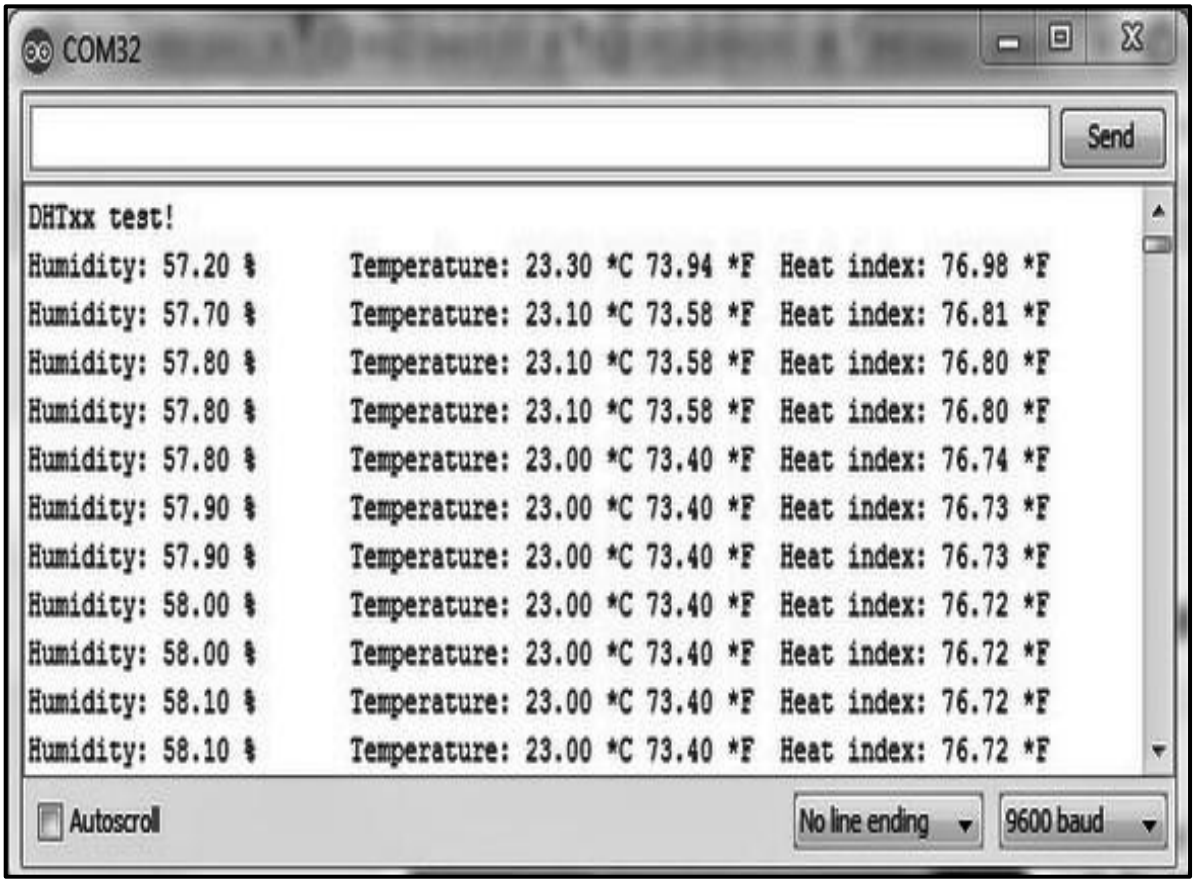


Рисунок 2.11 - Значения влажности в процентном соотношении и показатели температуры в «мониторе порта»

Чтобы *DHT11* и микроконтроллер работали корректно, их необходимо синхронизировать. Чтобы синхронизировать их, микроконтроллер посылает сигнал старта, который представляет собой импульс длительностью 20 мкс на выводе данных. После импульса, микроконтроллер ждет получения данных. В программе мы должны изменить направление вывода данных. На рисунке 2.12 представлена схема подключения датчика температуры и влажности *DHT11* к микроконтроллеру.

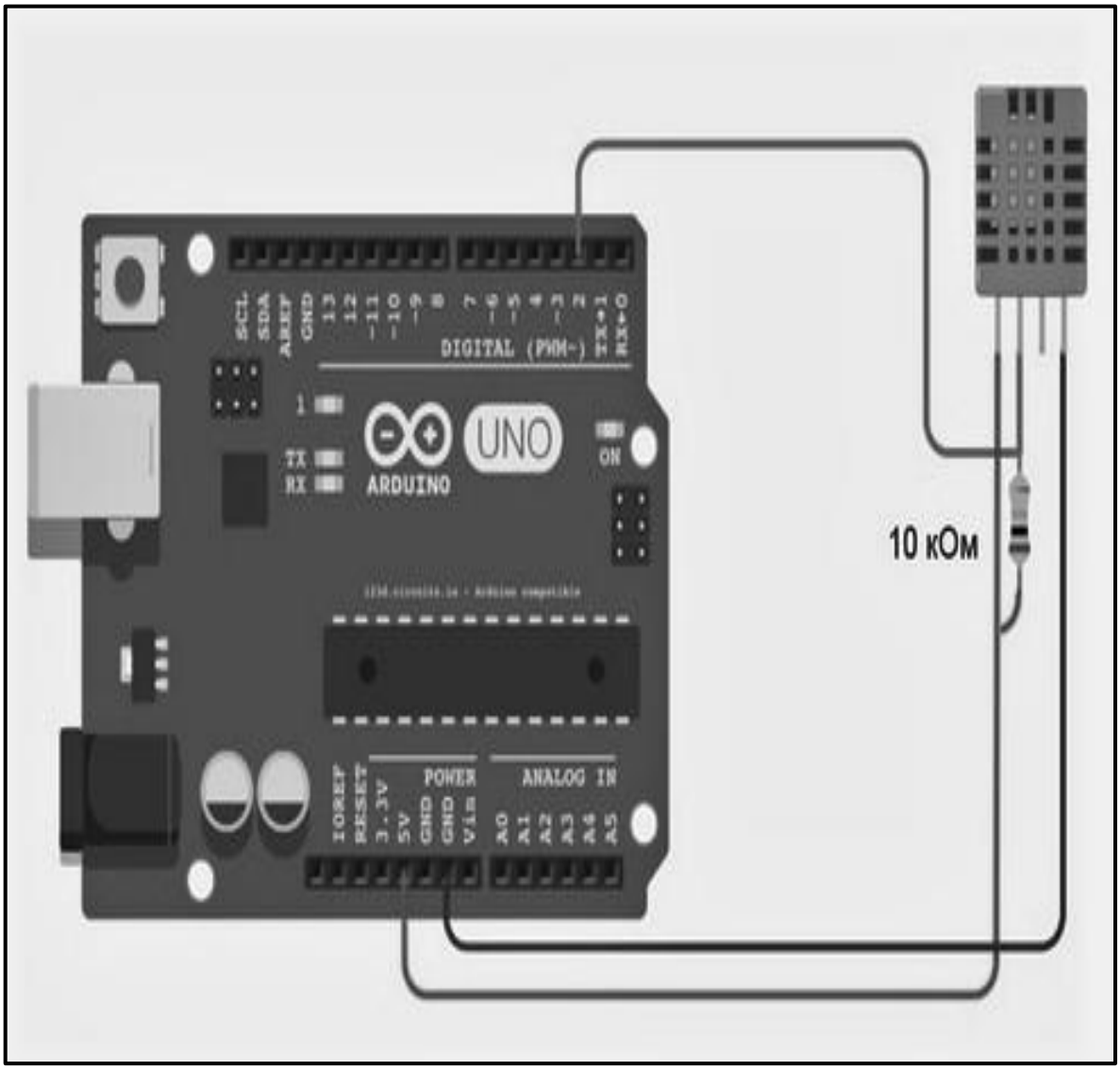


Рисунок 2.12 – Схема подключения датчика температуры и влажности *DHT11* к микроконтроллеру

Питание датчика от 3 В до 5,5 В. Если длина соединительного кабеля шины данных от датчика к микроконтроллеру не превышает 20 метров, то эту шину рекомендуется подтянуть к питанию резистором 5,1 кОм, если больше 20 метров – то другой подходящий меньший номинал. На рисунке 2.13 показана схема подключения «датчика дождя» для отечественных легковых автомобилей.

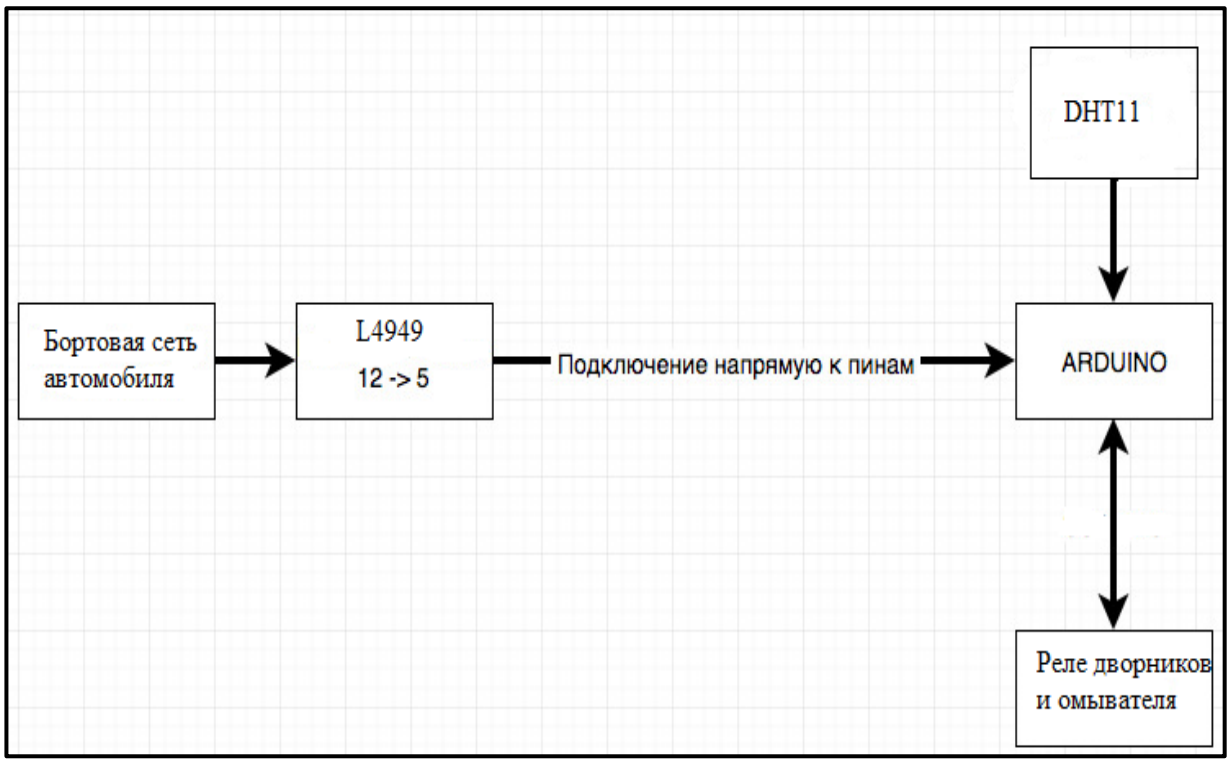


Рисунок 2.13 – Схема подключения «датчика дождя» для отечественных легковых автомобилей

Для связи с микроконтроллером датчик температуры и влажности *DHT11* использует однопроводный последовательный интерфейс. Один информационный обмен занимает около четырех мс и содержит: один бит запроса от микроконтроллера, один бит ответа датчика и 40 битов данных от датчика. В данные входят: 16 битов информации о влажности, 16 битов информации о температуре и восемь проверочных битов.

Формат обмена данными разделен на три этапа:

- инициализация;
- преамбула;
- передача данных.

Процесс чтения данных начинается с импульса инициализации который формирует микроконтроллер. Он должен установить на шине низкий уровень на время не менее 18 мс, для инициализации *DHT11*.

Микроконтроллер после формирования импульса инициализации должен сразу перевести порт в режим чтения (режим приема данных). Если датчик готов к передачи данных, он ответит сформировав преамбулу. Один период меандра составляет 160 мкс. На рисунке 2.14 представлен вид передачи данных полностью.

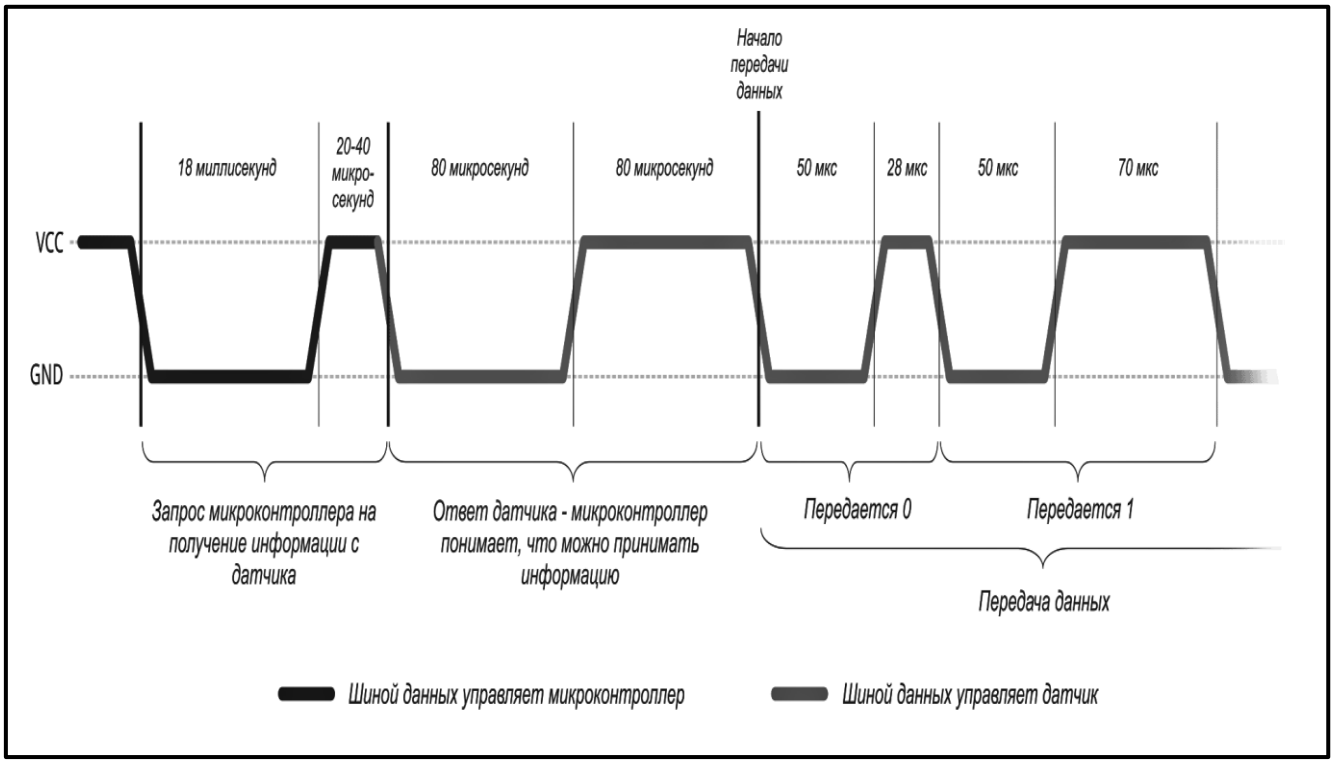


Рисунок 2.14 – Вид передачи данных полностью

Микроконтроллер инициирует передачу данных путем отправки сигнала «Старт». Для этой цели данный вывод МК должен быть сконфигурирован на выход. Микроконтроллер сначала переводит линию в состояние низкого уровня на 18 мс, затем переводит ее в высокое состояние на 20 - 40 мкс, прежде чем МК освобождает линию. Далее, датчик реагирует на сигнал «Старт» от микроконтроллера, отправив низкий уровень сигнала длительностью 80 мкс, а затем высокий уровень такой же продолжительностью. После обнаружения сигнала отклика от датчика, МК должен быть готов к приему данных от

датчика. Датчик посылает 40 бит данных, непрерывно в линию передачи данных. Следует отметить, что во время передачи байта, датчик посылает старший значащий бит первым. На рисунке 2.15 представлен сигнал «пуск» и ответ датчика.

Рисунок 2.15 – Сигнал «пуск» и ответ датчика

Для датчика *DHT11*, десятичные байты измерения температуры и влажности всегда равны нулю. Таким образом, первый и третий байт полученных данных фактически дают числовые значения измеренной относительной влажности (RH) и температуры (Temp.). Последний байт является байтом контрольной суммы, который используется, чтобы убедиться, что передача данных прошла без каких-либо ошибок. Если все пять байтов передаются успешно, то байт контрольной суммы должен быть равен последним восьми битам суммы первых четырех байтов. Передача данных идет по 1-Wire интерфейсу, однопроводной интерфейс, разработанный в конце 90-х годов фирмой *Dallas Semiconductor Corp.* Обмен информацией по шине 1-Wire идет следующим образом:

- обмен всегда ведется по инициативе одного ведущего устройства, которое в большинстве случаев является микроконтроллером;

- любой обмен информацией начинается с подачи импульса сброса в линию 1-Wire ведущим устройством;

- любое устройство, подключенное к 1-Wire после получения питания выдает в линию DQ импульс присутствия, называемый «Presence pulse». Этот же импульс устройство всегда выдает в линию, если обнаружит сигнал RESET;

- появление в шине 1-Wire импульса PRESENCE после выдачи RESET однозначно свидетельствует о наличии хотя бы одного подключенного устройства;

- обмен информации ведется так называемыми тайм-слотами: один тайм-слот служит для обмена одним битом информации;

- данные передаются побайтно, бит за битом, начиная с младшего бита. Достоверность переданных/принятых данных (проверка отсутствия искажений) гарантируется путем подсчета циклической контрольной суммы.

Чтобы отправить бит данных, датчик сначала переводит линию в низкий уровень на 50 мс. Если нужно отправить «0», датчик переводит линию в высокое положение на 26 - 28 мкс, или 70 мкс, если необходимо передать «1». Ширина положительного импульса несет в себе информацию о «0» и «1». На рисунке 2.16 представлена разниц в сигналах «0» и «1».

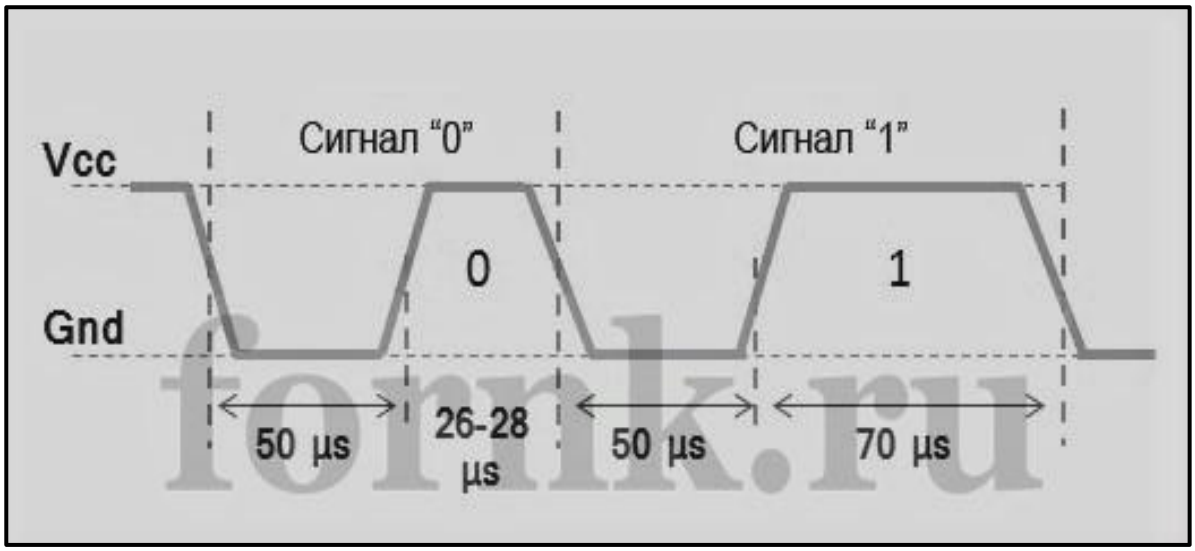


Рисунок 2.16 – Разница в сигналах «0» и «1»

В конце последнего переданного бита, датчик переводит линию передачи данных в низкое положение на 50 мс, а затем освобождает его. Для датчика *DHT11* требуется внешний подтягивающий резистор, который должен быть подключен между выводами Vcc и линии передачи данных, таким образом, в состоянии ожидания сигнал на линии данных всегда высокий. После окончания передачи данных и отпускании линии данных, датчик *DHT11* переходит в режим низкого энергопотребления, пока новый сигнал «Пуск» не поступит от микроконтроллера. На рисунке 2.17 представлена обобщенная картина обмена данными между микроконтроллером и датчиком *DHT11*.

Рисунок 2.17 - Обобщенная картина обмена данными

Микроконтроллеры *AVR* имеют три вида памяти:

1) память flash;

2) ОЗУ;

3) энергонезависимая память.

В микроконтроллере выделяется три адресных пространства в которых располагаются вышеперечисленные разновидности памяти. Памяти данных при приходится делить свое адресное пространство с ячейками памяти в которых хранятся регистры общего назначения и регистры ввода/вывода. Эти регистры физически не относятся к памяти данных, но находятся в том же адресном пространстве. Если начальные адреса памяти программ и энергонезависимой

памяти начинаются с нулевого адреса, то начальный адрес памяти данных не начинается с нулевого адреса – с нулевого адреса занимают места регистры общего назначения и регистры ввода/вывода, и только за ними следуют адреса ячеек памяти программ. Память программ (flash память) предназначена для хранения в ней программ, а также любых данных, которые не меняются в ходе выполнения программы (константы). При выключении питания микроконтроллера, все данные в памяти программ сохраняются. Оперативно-запоминающее устройство, оно же память данных типа SRAM, предназначена для хранения в ней различных данных, получаемых в результате работы программы. При выключении питания микроконтроллера, все данные хранящиеся в ней теряются. Энергонезависимая память также относится к памяти данных, но в отличие от последней имеет несколько особенностей. Предназначена она для хранения данных и констант, которые должны сохраняться при отсутствии питания. EEPROM имеют все микроконтроллеры. При выключении питания микроконтроллера все данные, хранящиеся в энергонезависимой памяти сохраняются (поэтому она и называется энергонезависимой). Программа для работы микроконтроллера, называемая прошивкой, представляет собой машинный код шестнадцатеричной системы счисления сохраненный в формате *.hex*. Файл состоит из текстовых ASCII строк. Каждая строка представляет собой одну запись. Каждая запись начинается с двоеточия (:), после которого идет набор шестнадцатеричных цифр кратных байту:

- начало записи (:);

- количество байт данных, содержащихся в этой записи. Занимает один байт (две шестнадцатеричных цифры), что соответствует 0-255 в десятичной системе;

- начальный адрес блока записываемых данных — два байта, этот адрес определяет абсолютное местоположение данных этой записи в двоичном файле. Один байт, обозначающий тип записи. Определены следующие типы записей:

0- запись содержит данные двоичного файла;

1- запись обозначает конец файла, данных не содержит, имеет характерный вид «:00000001FF»;

2- запись адреса сегмента;

3- запись расширенного адреса.

Байты данных, которые требуется сохранить в EPROM (их число указывается в начале записи, от 0 до 255 байт). Последний байт в записи является контрольной суммой. Рассчитывается так чтобы сумма всех байтов в записи была равна нулю. Строка заканчивается стандартной парой CR/LF (0Dh 0Ah). На рисунке 2.18 представлен пример прошивки микроконтроллера.

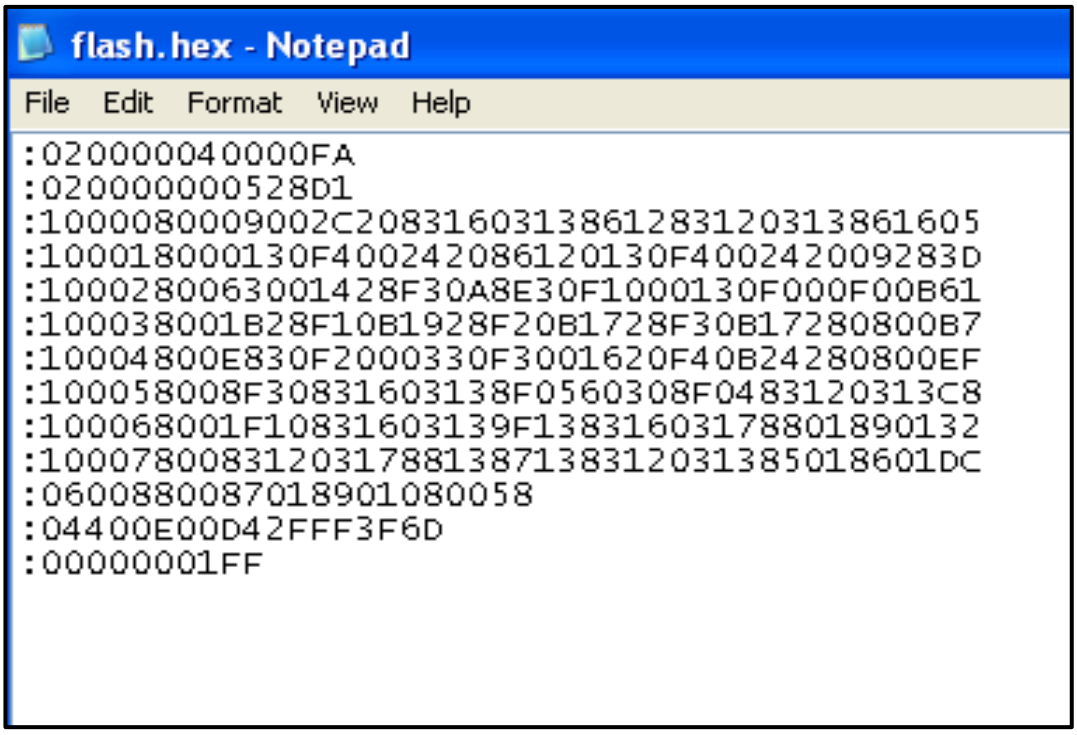


Рисунок 2.18 – Прошивка микроконтроллера

2.3 Вывод по главе 2

На основании анализа проведенного в первой главе, наилучшие показатели скорости работы у микроконтроллеров семейства *AVR*. Плата *Arduino Uno* не требует внешнего программатора, среда разработки *Arduino IDE* устанавливается на любой компьютер, выпущенный с 2000г. Плата *Arduino UNO R3* построена на базе микроконтроллера Atmega328 со следующими характеристиками:

- микроконтроллер *ATmega328*;
- рабочее напряжение 5 В;
- входное напряжение 7-12 В;
- 14 цифровых входов/выходов;
- 6 аналоговых входов;
- постоянный ток через вход/выход 40 мА;
- флеш-память 32 Кб;
- ОЗУ 2 Кб.

Плата *Arduino Uno* доступна для пользователя любого уровня. Плата *Arduino Uno* является общедоступной. Плата *Arduino Uno R3* продается на сайте *ru.aliexpress.com,* ее цена составляет 169 рублей. В качестве датчика выбран датчик влажности и температуры *DHT11* как наиболее простой и доступный. *DHT11* имеет следующие характеристики:

- питание 3-5 В;
- максимально потребляемый ток – 2,5 мА при преобразовании (при запросе данных);
- рассчитан на измерение уровня влажности в диапазоне от 20% до 80%. При этом точность измерений находится в диапазоне 5%;
- измеряет температуру в диапазоне от 0 до 50 градусов с точностью плюс-минус 2%;
- частота измерений не более 1 Гц (одно измерение в секунду).

Заказать датчик *DHT11* можно на сайте *ru.aliexpress.com*, стоимость которого составляет 50 рублей. Рассмотрен вид передачи сигнала от датчика к микроконтроллеру, разница между сигналами логического нуля или единицы. Плата *Arduino R3* может быть заменена на любую другую плату *Arduino UNO* на базе другого микроконтроллера.

ЗАКЛЮЧЕНИЕ

Данная бакалаврская работа направлена на разработку автоматизированной информационной системы для отечественных легковых автомобилей «датчик дождя».

В процессе выполнения бакалаврской работы реализованы следующие задачи:

- рассмотрены существующие автоматизированные системы отечественного автотранспорта;

- проведен обзор и анализ существующих семейств микроконтроллеров, рассмотрены среды разработок для каждого из них;

- разработана автоматизированная информационная система для отечественных легковых автомобилей «датчик дождя».

Спроектирована схема подключения датчика DHT11и платы Arduino Uno, к бортовой сети автомобиля. Подобран пятивольтовый регулятор напряжения для стабильной работы микроконтроллера от бортовой сети автомобиля. Рассмотрен вид передачи данных между датчиком влажности DHT11 и микроконтроллером. Рассмотрена разница между сигналами «нуля» и «единицы».

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1) Орлова И.Н. Электротехнические материалы: электротехнический справочник // Москва: Энергоиздательство 1985. – 488 с.

2) Белов Л.В. Самоучитель разработчика устройств на микроконтроллерах AVR. // Москва: Горячая линия, 2008. - 544с

3) Корабельников Е.А. Самоучитель по программированию PIC микроконтроллеров. // Москва: МДК АйТи, 2008 г. – 279с.

4) Яценкоп В. С. Микроконтроллеры Microchip с аппаратной поддержкой USB. // Москва: МДК Пресс. 2008 г. - 400с.

5) Тавернье А.К. PIC-микроконтроллеры. Практика применения. // Москва: ДМК Пресс 2002 г. - 272с.

6) А. Кёниг и М. Кёниг Полное руководство по PIC-микроконтроллерам PIC18, PIC10F. // Москва ДМК АйТи. 2007г. - 255с.

7) Ревич Ю. В. Практическое программирование микроконтроллеров Atmel AVR на языке ассемблера. // БХВ-Петербург 2008г. – 384с.

8) Трамперт В. Измерение, управление и регулирование с помощью AVR-микро-контроллеров. // Москва МК-Пресс 2006г. – 192с.

9) Уилмсхерст Т. Разработка встроенных систем с помощью микроконтроллеров PIC. // Москва МК-Пресс 2008 г. – 544с.

10) Кравченко А. В. 10 практических устройств на AVR-микроконтроллерах. Книга 1. // Москва МК-Пресс 2008г. – 224с.

11) Э. Парр. Программируемые контроллеры: руководство для инженера. // Москва М Бином 2007г. – 516с.

12) Мортон Д. Микроконтроллеры AVR. Вводный курс. // Москва Додэка-XXI 2006 г. – 270с.

13) М. С. Голубцов. Микроконтроллеры AVR: от простого к сложному. // Санкт-Петербург БХВ. 2003г. – 288с.

14) Редькин П. П. 32/16-битные микроконтроллеры ARM7 семейства AT91SAM7 фирмы Atmel. // Москва Додэка-XXI 2008г. – 704с.

15) Хартов В.Я. Микроконтроллеры AVR. Практикум для начинающих. // Москва МГТУ им Баумана 2007г. – 240с.

16) Белов А. В. Создаем устройства на микроконтроллерах. // Москва Наука и техника 2007г. – 304с.

17) Евстифеев А.В. Микроконтроллеры AVR семейств Tiny и Mega фирмы «ATMEL». // Москва Издательский дом «Додэка XXI», 2004г. - 426с.

18) Баранов В.Н. Применение микроконтроллеров AVR: схемы, алгоритмы, программы. // Москва Додэка 2004г. – 288с.

19) Фред И. Сетевой и межсетевой обмен данными с микроконтроллерами. // Москва Додэка-XXI Серия: Программируемые системы. 2007г. – 376с.

20) Катцен С. Всё, что вам необходимо знать о PIC микроконтроллерах. // Москва Додека-XXI 2006г. – 651с.

ПРИЛОЖЕНИЕ А

Слайды презентации

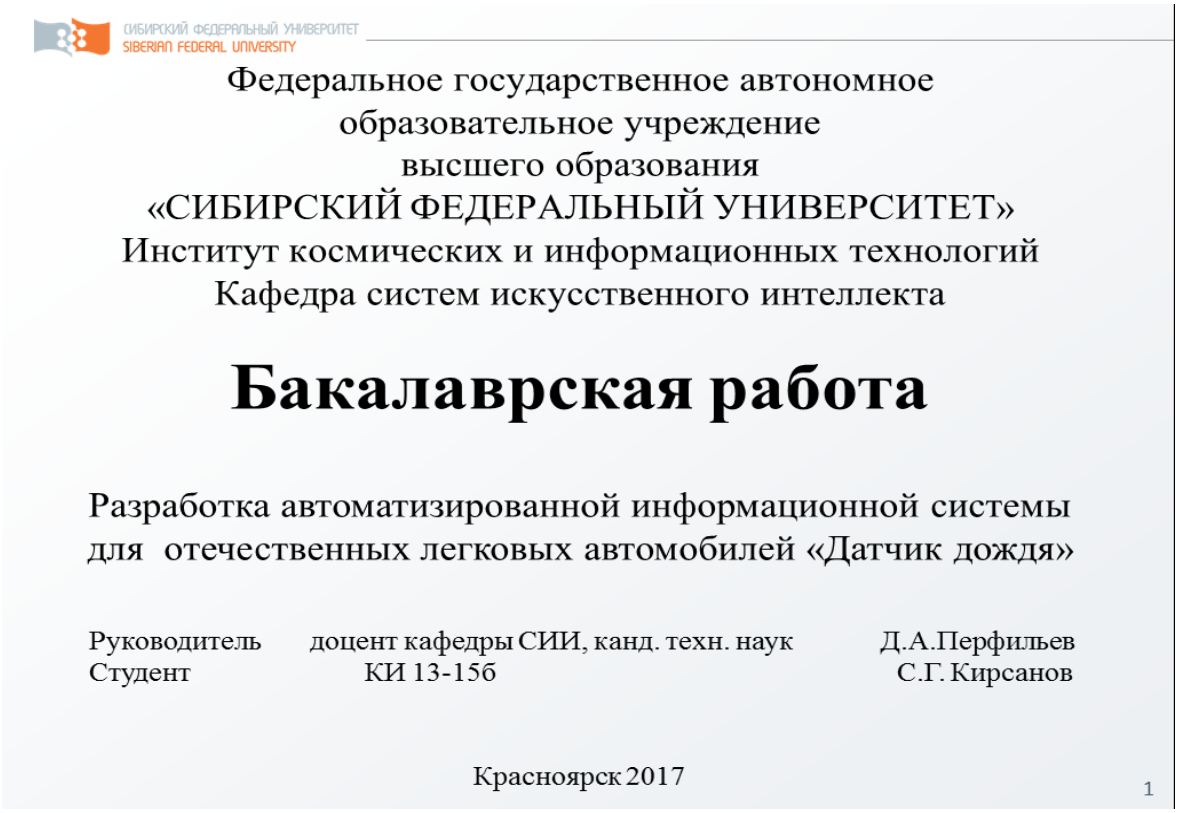


Рисунок А.1 – Титульный лист

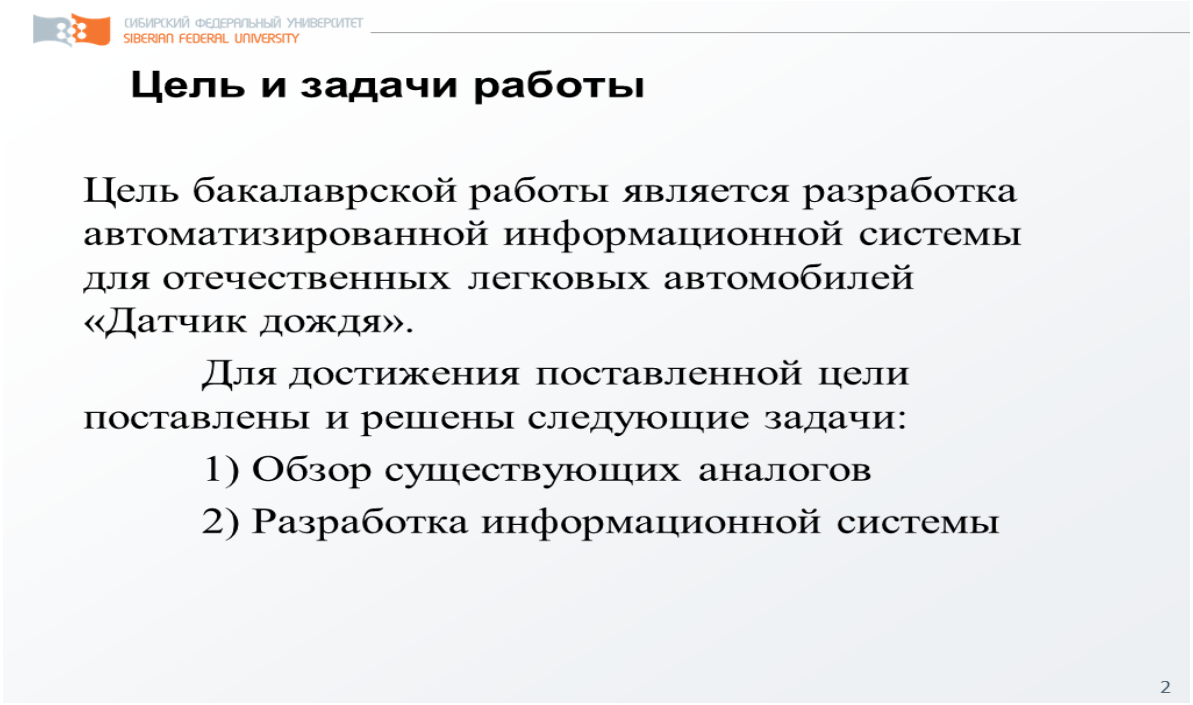


Рисунок А.2 – Цель и задачи

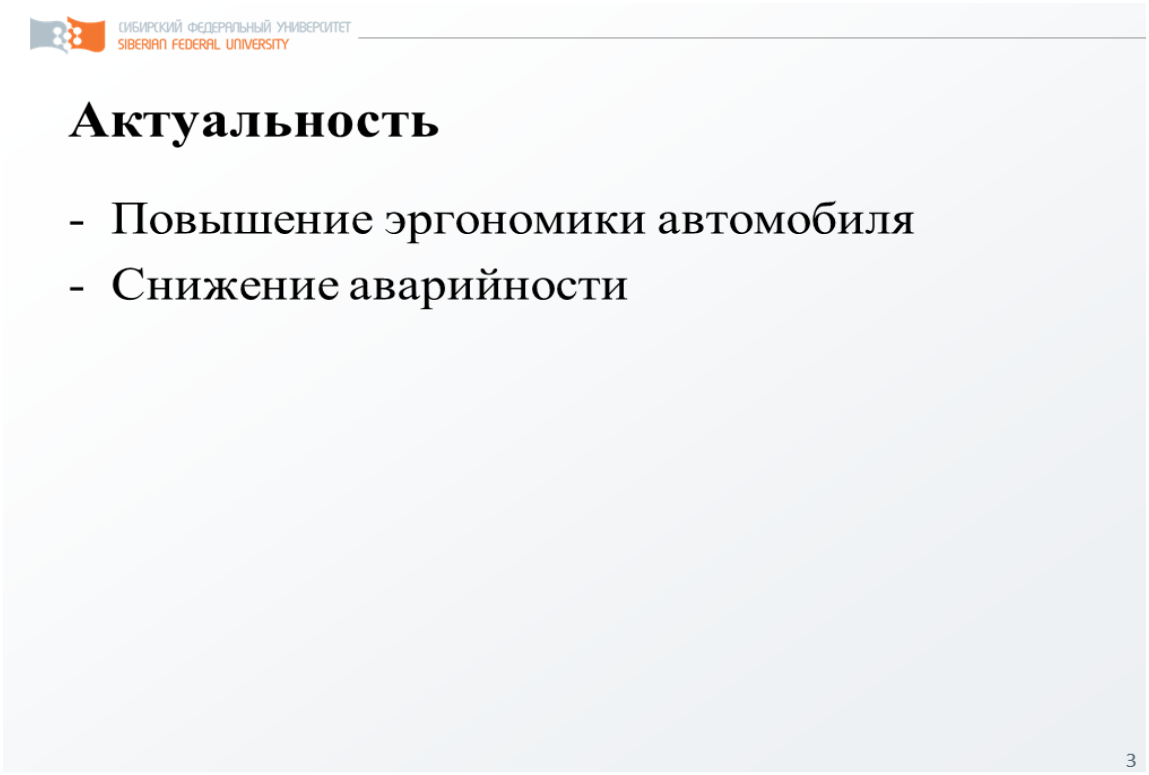


Рисунок А.3 – Актуальность

Рисунок А.4 – Интерфейс CAN фирмы BOCSH

Рисунок А.5 – АМК-211501

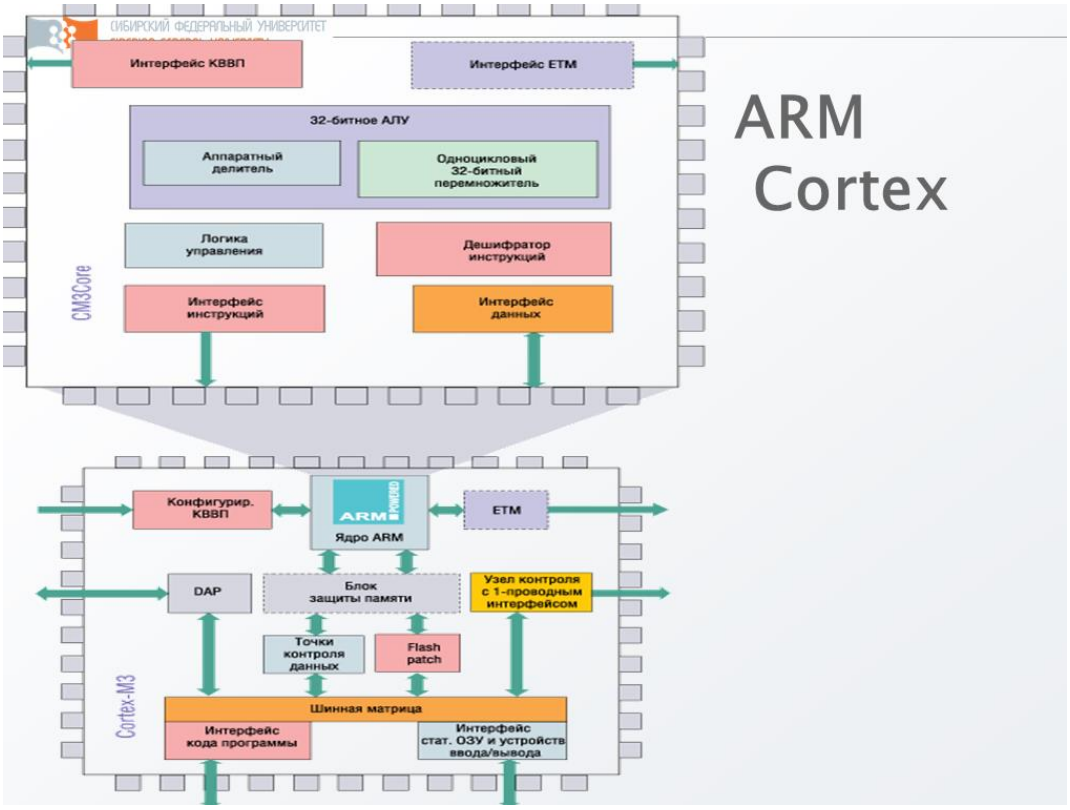


Рисунок А.6 – микроконтроллеры ARM Cortex

Рисунок А.7 – PIC микроконтроллеры

Рисунок А.8 – AVR микроконтроллеры

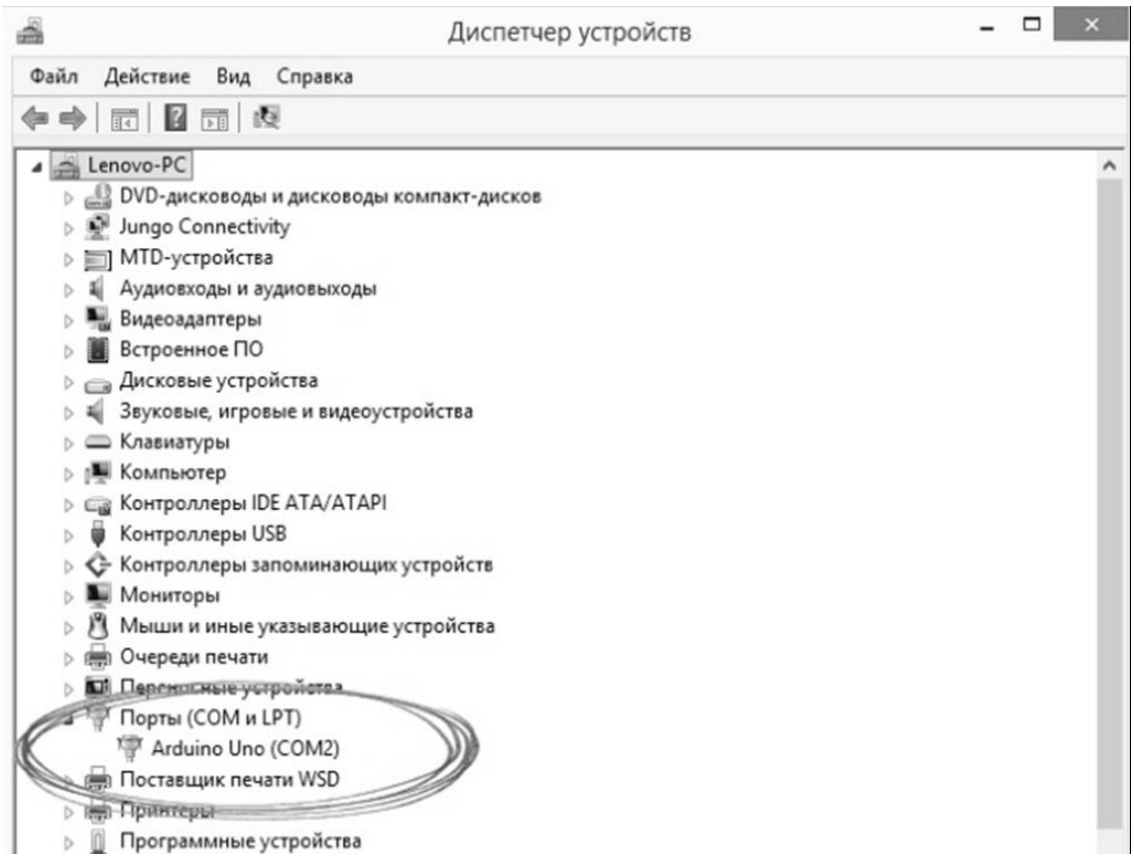


Рисунок А.9 - Диспетчер устройств, вкладка «Порты»

Рисунок А.10 – Структурная схема датчика дождя

Распределение пинов микроконтроллера и код совместной работы щеток и омывателя

```
 9   ;       GP0 (7) - вход управления периодом включения реле дворников.
10   ;       GP1 (6) - выход включения реле щеток.
11   ;       GP2 (5) - выход включения реле омывателя.
12   ;       GP3 (4) - вход включения режима автомата.
13   ;       GP4 (3) - вход включения прерывистого режима.
14   ;       GP5 (2) - выход включения светодиода.
```

```
74          bsf     GPIO, 5             ; Включаем светодиод.
75          bsf     GPIO, 2             ; Включаем реле омывателя.
76          call    Delay_1.0
77          call    Delay_0.6
78          bsf     GPIO, 1             ; Включаем реле дворников.
79          call    Delay_1.0
80          call    Delay_1.0
81          call    Delay_0.6
82          bcf     GPIO, 2             ; Выключаем реле омывателя.
83          call    Delay_1.0
84          call    Delay_1.0
85          call    Delay_1.0
86    ;     call    Delay_1.0
87          bcf     GPIO, 1             ; Выключаем реле дворников.
88          bcf     GPIO, 5             ; Выключаем светодиод.
89          return
```

Рисунок А.11 – Код совместной работы щеток и омывателя

Плата Arduino Uno

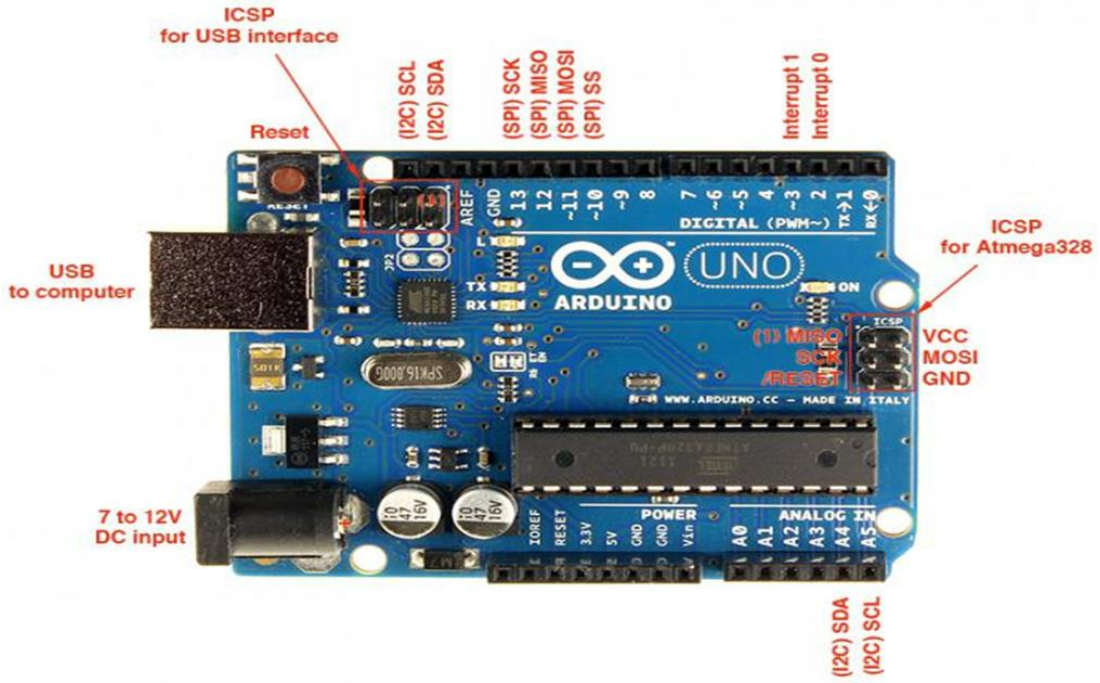


Рисунок А.12 – Плата Arduino Uno

Рисунок А.13 – Регулятор напряжения L4949

Рисунок А.14 – Схема подключения датчика к микроконтроллеру

СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ
SIBERIAN FEDERAL UNIVERSITY

Инициализация DHT11 и считывание данных

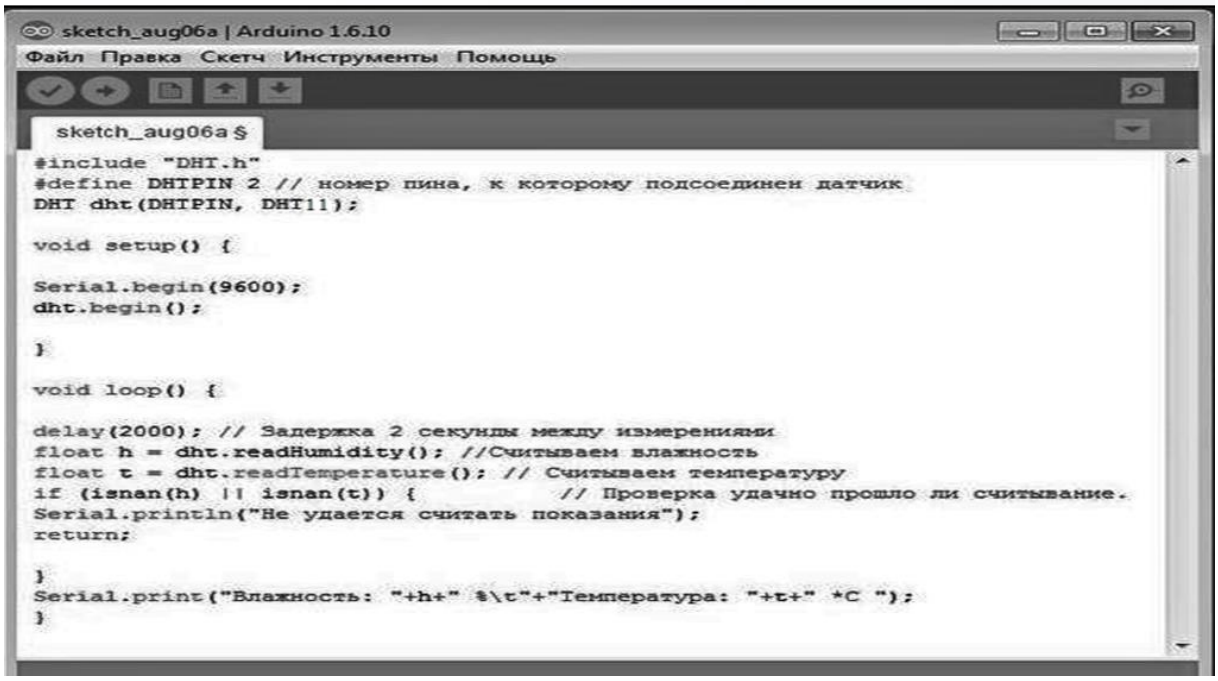


Рисунок А.15 – Инициализация DHT11 и считывание данных

СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ
SIBERIAN FEDERAL UNIVERSITY

Итоговая схема подключения

Рисунок А.16 – Итоговая схема подключения

Рисунок А.17 – Вид передачи данных

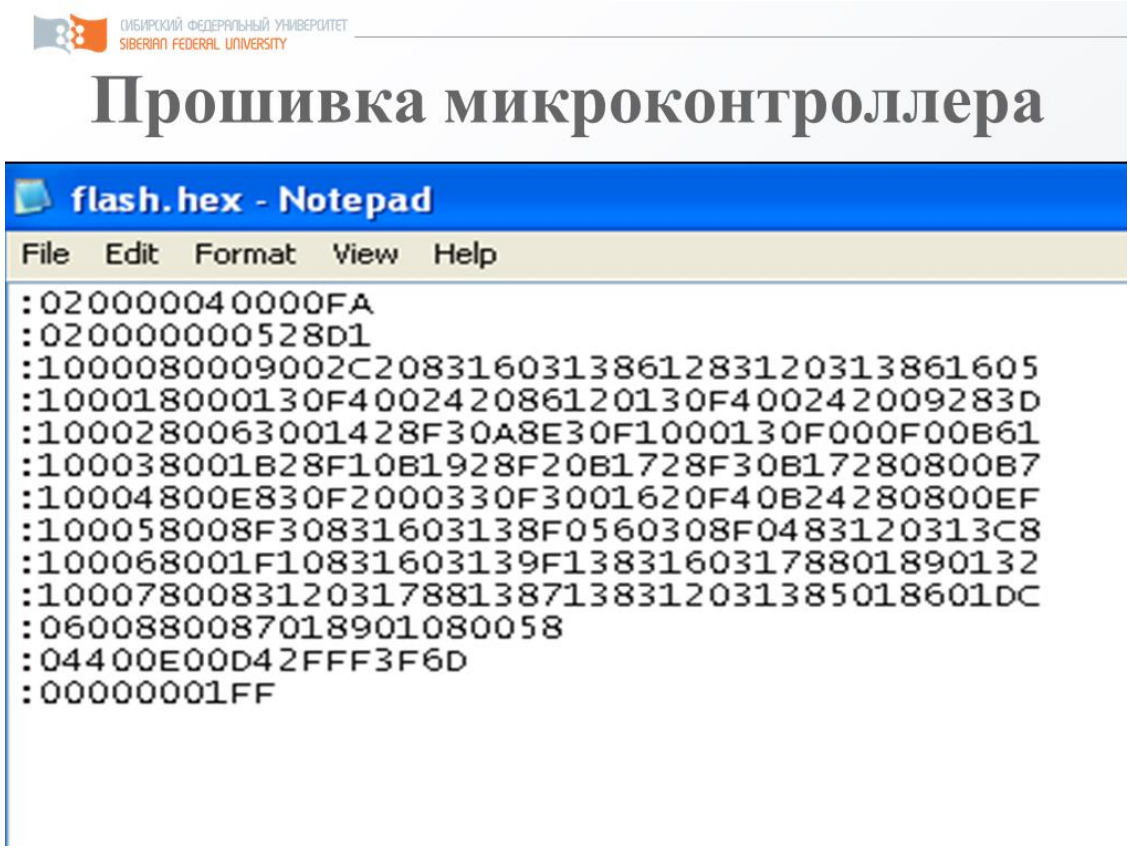


Рисунок А.18 – Прошивка микроконтроллера

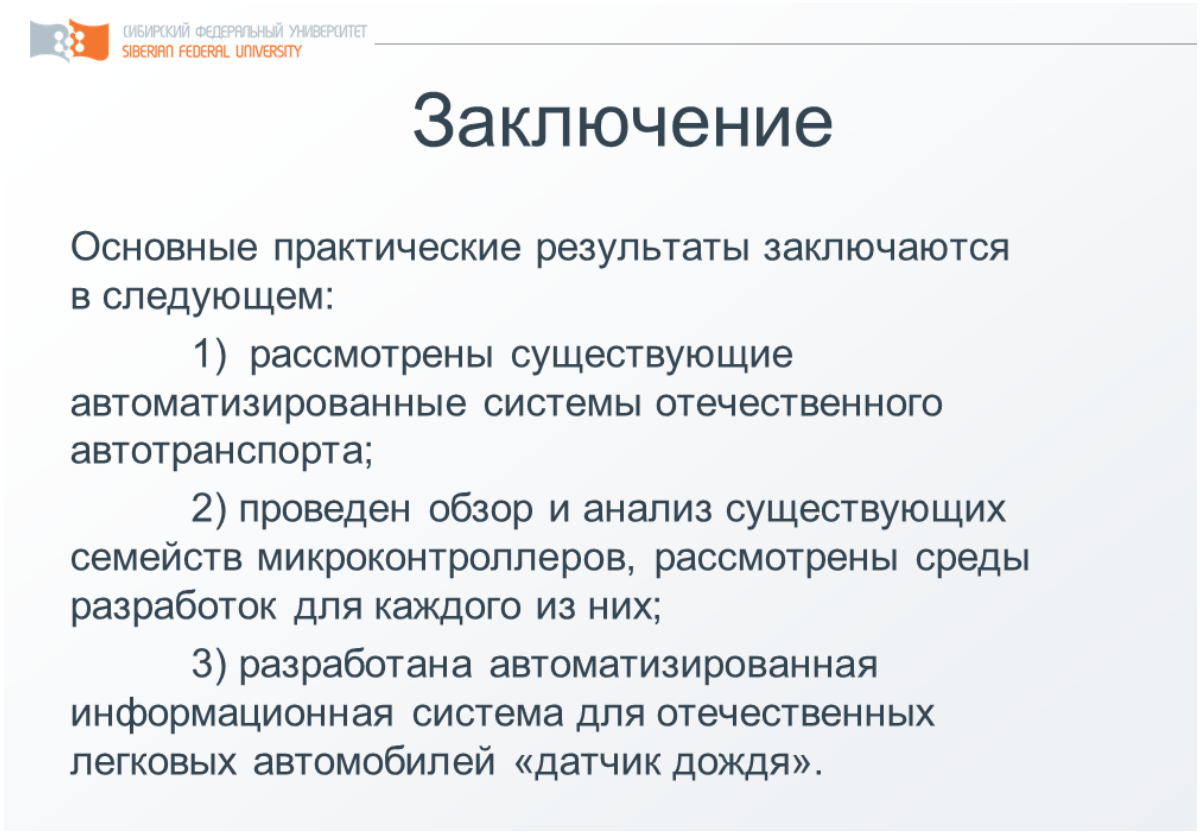


Рисунок А.19 - Заключение